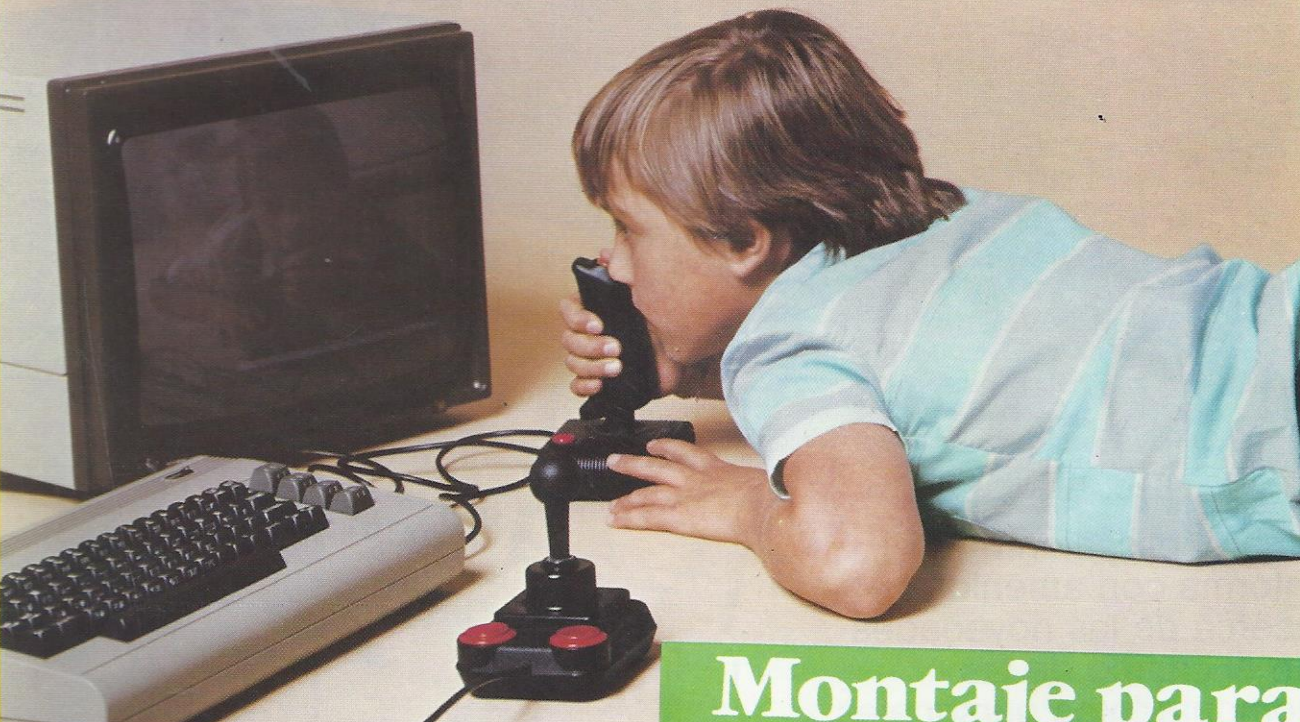
commodore Magazine

AÑO I - Núm. 8 - Octubre 1984 - 250 Ptas.

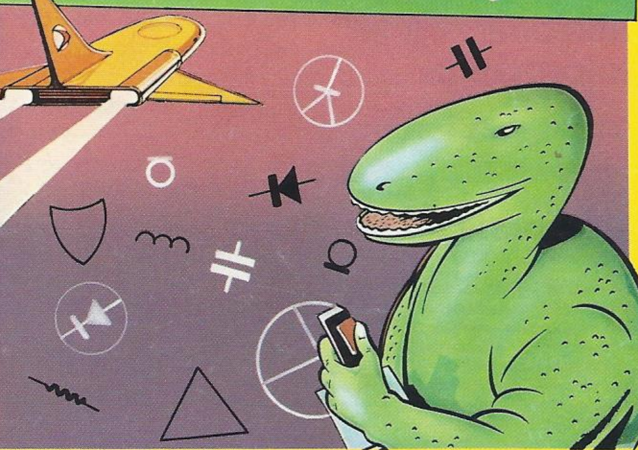
REVISTA INDEPENDIENTE PARA USUARIOS

Todo sobre Joysticks y Paddles



Montaje para jugar mejor





Logo para el 64

commodore Magazine

Sumario

DIRECTOR:
Alejandro Diges
COORDINADOR EDITORIAL:
J. A. Sanz
REDACCION:
Roberto Menéndez
Anibal Pardo
Teresa Aranda
Gumersindo García
Fernando García
DISEÑO:
Ricardo Segura

Editada por
PUBLINFORMATICA S.A.
PRESIDENTE:
Fernando Bolín
DIRECTOR EDITORIAL
Norberto Gallego

Administración
INFODIS S.A.
CONSEJERO DELEGADO:
Fernando Bolín
GERENTE DE CIRCULACION Y VENTAS
Luis Carrero
PRODUCCION
Miguel Onieva
SERVICIO CLIENTES
Antonio Zurdo
JEFE DE PUBLICIDAD
María José Martín

Dirección:
Redacción y administración
C/ Bravo Murillo, 377.
28020 Madrid
Tel. 733 73 13

Publicidad Madrid
C/ Bravo Murillo, 377.
28020 Madrid
Tel. 733 96 62/96

Publicidad Barcelona
C/ Pelayo 12. 08001 Barcelona
Tel. (93) 301 47 00 ext. 27

Depósito Legal: M-6622-1984.
Distribuye: S.E.G.L.
Avda. Valdelaparra, s/n.
Alcobendas. Madrid.

Fotomecánica: Karmat
C/ Pantoja, 10. Madrid.
Imprime: Novograph, S.A.
Carretera de Irún, km. 12,450.
Madrid.
Solicitado control O.J.D.

Año 1
Num. 8

SUSCRIPCIONES

Rogamos dirija toda la correspondencia relacionada con suscripciones o números atrasados a:
COMMODORE
MAGAZINE
EDISA
LOPEZ DE HOYOS, 141
5.º
28002-MADRID

7 Joystick y Paddle para todos. Estos útiles accesorios para los ordenadores personales **Commodore 64** y **Vic 20** son estudiados a fondo para su mejor aprovechamiento.

12 Montaje de Joysticks. Un caso práctico para que el usuario pueda jugar y alcanzar el máximo rendimiento frente a su ordenador.

16 Concurso. Siguen llegando nuevos programas a nuestra redacción. Cada vez es más difícil hacer la selección para publicar, llegan más y mejores que en un principio. En esta ocasión: Memory, Laberinto de Picas y Gráficos en tres dimensiones.

28 Segunda parte del Misterio del BASIC. Un lenguaje absolutamente imprescindible para dominar el manejo de los **Commodore**. Paso a paso, describimos como es el lenguaje y sus aplicaciones.

32 Comentarios. Dos programas comentados por nuestro especialista Fernando García: TANX y MOON BUGGGY.

34 El LOGO. Lenguaje desarrollado en América en la década de los sesenta y que hasta 1981 no se ha aplicado a las necesidades de los personales. **Commodore** ha hecho una excelente versión del **LOGO** de la que les contamos algunos secretos.

40 Cálculo financiero. Para todos aquellos que tengan problemas con las cuentas de casa o con Hacienda, este programa y sus explicaciones, le vendrá muy bien.

44 Programas. La Cucaracha cósmica, Simón-64 y el Piloto de las galaxias.

Esta revista no mantiene relación de dependencia de ningún tipo con respecto de los fabricantes de ordenadores Commodore Business Machines ni de sus representantes.

Editorial

Cuando estábamos planificando la presente edición de la revista, evaluamos los diversos temas con los que contábamos para figurar en portada. Al final nos inclinamos por los *joysticks* y *paddle*, dispositivos de fundamento sencillo que todo usuario de ordenador personal se ha planteado alguna vez adquirir. El tema es extenso e interesante, por lo cual decidimos trocearlo en dos partes. La siguiente se publicará en el próximo número, también en las dos vertientes adoptadas: teórica y práctica. Hoy publicamos la fabricación de un *joystick* artesanal, dentro de un mes haremos lo propio con un *paddle*.

Los sufridos lectores usuarios del *Vic 20* habrán podido observar algunos desequilibrios en el reparto de los programas publicados, en favor del *Commodore 64*. Esto vino impuesto por la calidad de los programas que teníamos en cartera, lo cual no quiere decir que en determinados números, por ejemplo, el siguiente, la balanza se incline en sentido contrario. En esa ocasión serán los usuarios del *C-64* quienes hagan surgir sus reclamaciones. Pero así es la vida, nunca llueve a gusto de todos, por más que nosotros nos empeñemos. De todas maneras, ya veréis el día que nos sea posible disponer de mayor número de páginas.

Desde aquí deseamos aprovechar para animaros a que sigáis enviando el fruto de vuestros trabajos, en forma de programa. Por nuestra parte prometemos analizar todos los envíos, aunque lleguemos al límite de nuestra capacidad de desbordamiento. Este último factor es el que nos impide contestar puntualmente a vuestras interesantes y amables cartas de ánimo y dudas. Os rogamos un poco de paciencia al respecto. Nuestra intención es contestarles todas, bien en la sección de cartas o personalmente. Otras veces agrupamos un número de cartas, por ser similar su contenido y, aunque contestamos a un solo lector, la respuesta va dirigida a todos los demás. No os enfadéis si la respuesta no llega con la celeridad que os gustaría y, por favor, una reflexión: si nos dedicáramos a responderlas todas inmediatamente. ¿Quién escribiría la revista?

Próximamente habrá novedades, una de ellas consistirá en explicar más detalladamente los programas que se publiquen.

Ahora os dejamos que disfrutéis la revista y únicamente recomendaros que no dejéis de leer el fantástico artículo sobre el lenguaje *LOGO* que ha redactado un colaborador muy experto en la materia.



Códigos de control para el VIC-20 y el C-64

COMO SE VE	COMO SE TECLEA	EFECTO CONSEGUIDO
COLORES DEL VIC-20 Y DEL 64		
	CTRL+1	NEGRO
	CTRL+2	BLANCO
	CTRL+3	ROJO
	CTRL+4	CIAN
	CTRL+5	PURPURA
	CTRL+6	VERDE
	CTRL+7	AZUL
	CTRL+8	AMARILLO
COLORES DEL 64 SOLAMENTE		
	CBM+1	NARANJA
	CBM+2	MARRON
	CBM+3	ROSA
	CBM+4	GRIS OSCURO
	CBM+5	GRIS MEDIO
	CBM+6	VERDE CLARO
	CBM+7	AZUL CLARO
	CBM+8	GRIS CLARO
CODIGOS DE CURSOR Y CONTROL		
	HOME	CURSOR A CASA
	SHIFT+HOME	LIMPIA PANTALLA
	CRSR	CURSOR IZQUIERDA
	SHIFT+CRSR	CURSOR DERECHA
	CRSR	CURSOR ABAJO
	SHIFT+CRSR	CURSOR ARRIBA
	CTRL+9	CARACTER INVERSO
	CTRL+0	CARACTER NORMAL
	DEL	BORRAR
	SHIFT+DEL	INSERTAR
TECLAS DE FUNCION		
	F1	
	F2=SHIFT+F1	
	F3	
	F4=SHIFT+F3	
	F5	
	F6=SHIFT+F5	
	F7	
	F8=SHIFT+F7	

¡¡Exclusivo para lectores de

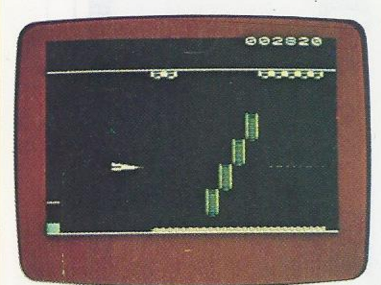






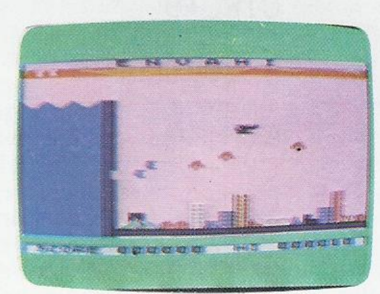
Ha seleccionado para usted cuatro excelentes programas de juego para su ordenador **VIC-20.**

Diviértase con ellos y, **además, ahorre dinero** con esta oferta única y exclusiva para los lectores de esta revista.



CREEPERS

Controlas al último guerrero que defiende la alimentación de energía de las ciudades subterráneas. Los invasores se agarran al tejado, encima de tí, y procuran controlar los bloques de energía. Si lo logran, los bloques de una zona se convierten en mutantes y se alojan en ti. Los nervios y los reflejos son vitales. Te estás jugando el todo o nada.
Precio: 1.350 ptas.

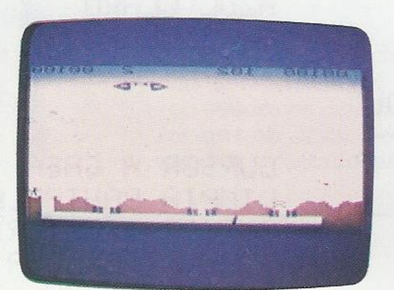


ENVAHI

Arriba, sobre la ciudad de Envahi, está la presa que alimenta de energía y agua a sus habitantes. Tú estás controlando un helicóptero armado que patrulla el cielo para protegerla de los truhanes que vienen decididos a destruirla. No es todo: el velero comilón cruza desde el este y de unos pocos mordiscos puede deshacer la presa e inundar la ciudad. Ah ¡No olvides tu paraguas; puede ser útil!
Precio: 1.350 ptas.

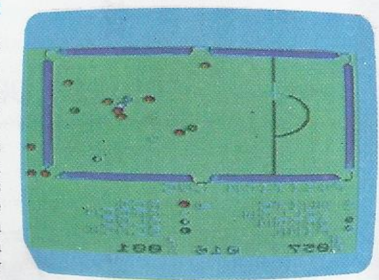
MISSION MERCURY

La Agencia Tierra Espacio acaba de anunciar que se ha perdido toda la vida en Mercurio, cuando se recibe un dramático SOS desde el planeta. Como eres el astronáuta más experimentado, te escogen para organizar la misión de rescate. "Estación Mercurio llamando al equipo de rescate... fábrica de caminos para lanzaderas... desestabilizado... aterrizar solamente una vez...". De prisa, no tienes tiempo que perder.
Precio: 1.350 ptas.



SNOOKER

No confundirse. Snooker es sólo aparentemente un juego de billar. En la práctica es mucho más, un medio para desarrollar el concepto y la visión del espacio. Pocos programas son tan adecuados como éste para adquirir la coordinación de velocidad, ángulos e intersecciones. Usted puede tomárselo como quiera: como un billar (¿por qué no?) o como una clase práctica de física y geometría aplicadas.
Precio: 1.350 ptas.



Oferta especial 4X3

Usted puede comprar cualquiera de estos cassettes a su precio individual de 1.350 Ptas. Pero también puede adquirir el lote de 4 por sólo 4.050 Ptas. y AHORRARSE 1.350 Ptas. O sea, cuatro cassettes por el precio de tres. Aproveche esta oferta limitada.

Recorte y envíe este cupón HOY MISMO a:
INFODIS, S.A.
Bravo Murillo, 377-5.º-A
28020 MADRID

CUPON DE PEDIDO

Envíenme a mi domicilio, al precio de 1.350 Ptas. cada una, más 95 Ptas. de gastos, la cassette o cassettes que le indico a continuación marcando con una (X) en el casillero correspondiente. Queda bien entendido que pidiendo las 4 cassettes obtengo un ahorro de 1.350 Ptas., lo que me supone UNA CASSETTE GRATIS (**Sólo pagaré 4.050 Ptas. más 95 Ptas. por gastos de envío**).

☐ **CREEPERS** ☐ **ENVAHI** ☐ **MISSION MERCURY** ☐ **SNOOKER**

☐ Las 4 cassettes, en las ventajosas condiciones de AHORRO para mí (**4.050 Ptas.**).

El importe lo abonaré:
POR CHEQUE ☐ **CONTRA REEMBOLSO** ☐ **CON TARJETA DE CREDITO** ☐
American Express ☐ **Visa** ☐ **Interbank** ☐ Fecha de caducidad ____________

Número de mi Tarjeta ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

Nombre ____________

Dirección ____________

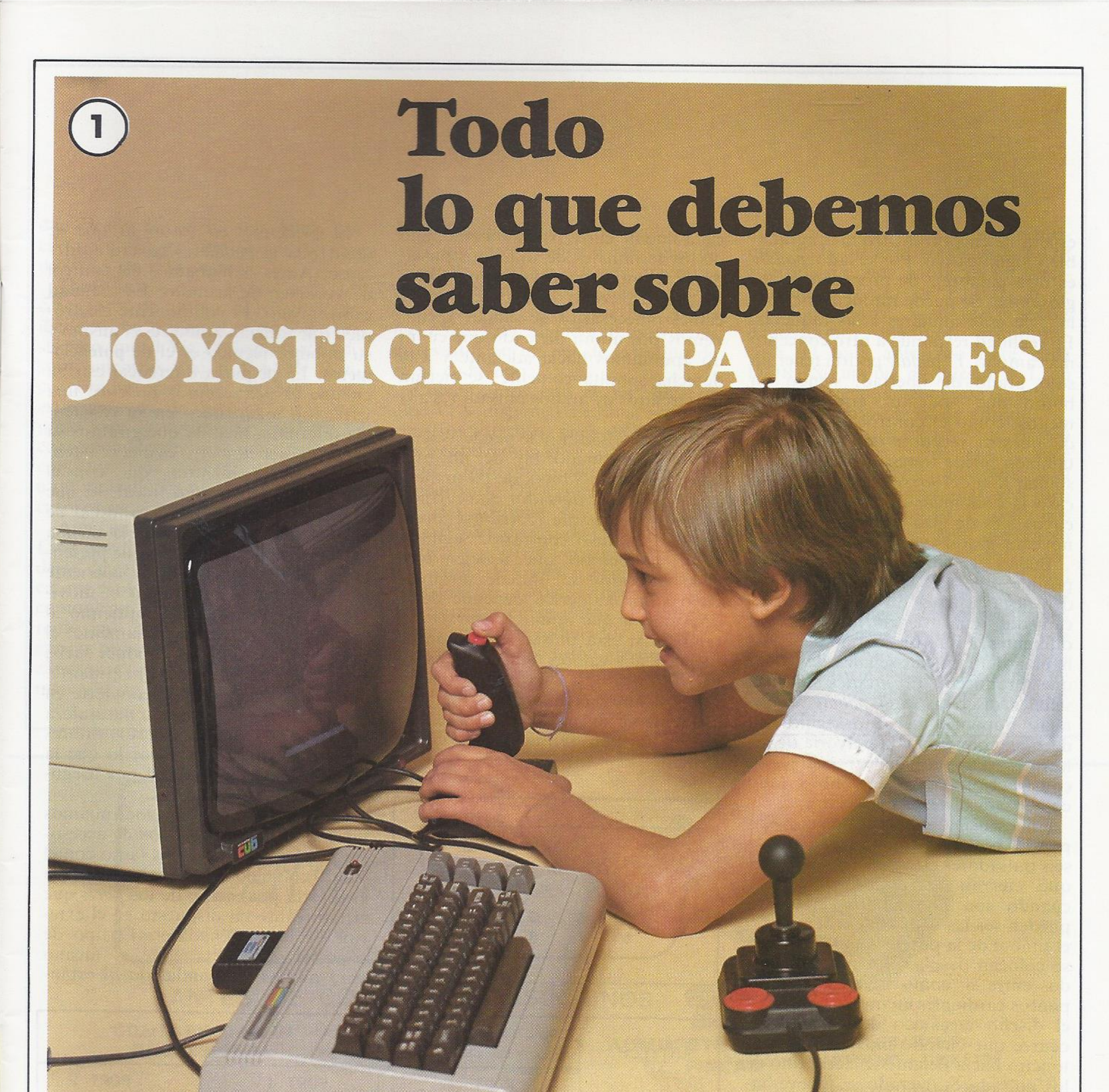
Ciudad ____________ D.P. ______

Provincia ____________

Cada envío llevará 95 Ptas. en concepto de gastos.

1

Todo lo que debemos saber sobre JOYSTICKS Y PADDLES



Los juegos representan uno de los mayores atractivos que nos incitan a comprar un ordenador personal. No conviene olvidar otras posibilidades, tales como el *software* educativo y la posibilidad de aprender lenguajes de programación, principalmente BASIC. Como es lógico, cualquier juego que se precie tiene una característica común y fundamental: la interactividad. Es difícil imaginar un juego con el que el ordenador sea juez y parte, sin permitir la intervención del asombrado usuario. De momento están lejos las maravillas que el futuro nos promete, tales como interactividad por medio de la palabra, donde el ordenador reconoce y sintetiza la voz. Hoy por hoy el tipo de accesorio más ampliamente utilizado es el *joystick* seguido por el *paddle*.

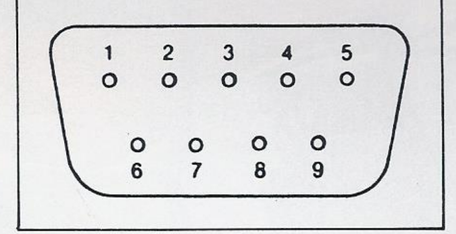
Comencemos por recordar qué es cada una de estas dos cosas que tan bien suenan. *Joystick* viene a significar literalmente "bastoncillo para jugar", y eso es en definitiva un bastón similar a una réplica en miniatura de la palanca de cambio para un automóvil, que nos permite ejercer un cierto control sobre lo que sucede en el interior de la máquina. En este caso, el programa que está corriendo el ordenador debe disponer de órdenes adecuadas que permiten actuar al *joystick*. Por el momento seguiremos utilizando el vocablo inglés, a la

espera de encontrar otro más agradable y significativo. Por su lado, *paddle* es otra palabreja de procedencia anglosajona. Se puede traducir por "pala", de esas similares a las raquetas. En realidad responde a otro concepto más amplio. El *paddle* podría tener un solo grado de libertad, siendo un botón que podemos girar más o menos, como el control de volumen de nuestro receptor de radio. Pero también puede tomar una apariencia similar a la del *joystick*, siendo en tal caso la combinación de dos *paddles*, cuyos ejes de giro están desplazados noventa grados entre sí.

Bueno, se preguntará, **¿cómo** es posible que dos accesorios que pueden llegar a tener un aspecto prácticamente igual tienen nombres tan **diferentes?** Es ahora cuando debemos hablar del tipo de señales de salida que producen. El *joystick* nos da señales discretas y el *paddle*, continuas. Supongo que esto seguirá sin decirles gran cosa a la mayoría de los lectores. Después de ver cuál es la construcción física de ambos todo comenzará a verse más claro.

En esencia, el *joystick* está compuesto por un conjunto de cinco o seis pulsadores eléctricos, dispositivos que cierran un circuito eléctrico cuando son presionados, tal como pueden ser las teclas del ordenador o el timbre de la puerta. Cuatro de ellos se colocan desplazados noventa grados entre sí, como serían los cuatro puntos cardinales en una brújula. Por el diseño mecánico del dispositivo, ocurre que cuando movemos el bastoncito hacia delante se cierra uno de los pulsadores, cuando se mueve hacia detrás se activa el situado inmediatamente enfrente y lo mismo ocurre para los movimientos a izquierda y derecha. Los otros dos pulsadores, llamados de fuego, son activados directamente por el usuario a voluntad, como es el caso de abrir el fuego láser contra los voraces marcianitos. Son dos porque suelen situarse en diferentes lugares del *joystick*, de cara a eliminar las posibilidades de que el marciano nos pille desprevenidos. Pues bien, el cierre de cada uno de estos pulsadores produce una señal, que será circuito abierto o circuito cerrado, o si lo preferimos 1 y 0, en los cables que están conectados a cada uno de ellos. Normalmente, uno de los terminales de todos los pulsadores va unido a otro común a todos los demás, como forma de ahorrar cables innecesarios de salida. Terminemos por ahora diciendo que cuando apuntamos hacia adelante y a la derecha simultaneamente, se cierran los circuitos de ambos pulsadores a la vez.

A cada pulsador se le puede asociar a un determinado bit de un byte (agrupación de 8 bits), de tal manera que cuando ese pulsador está cerrado su bit asociado vale 1, atribuyéndosele un 0 en caso contrario. Según el valor ponderado que asociemos a cada pulsador se obtiene un byte, cuyo valor numérico depende de la dirección a que apunte el *joystick* y también si está pulsado el botón de fuego. Estos valores nunca son distintos de un grupo de valores "discretos", siendo las diferencias lo suficientemente dispares como para que no exista una sensación de continuidad.



El principio en torno al que se desarrolla el *paddle* es bastante diferente. **Antes** hablábamos del control de volumen de la radio. En realidad es lo mismo. El componente electrónico que utilizan ambos es el mismo y su nombre técnico es el de **potenciómetro.** La pequeña diferencia estriba en que el del receptor produce una variación logarítmica en su resistencia interna, a medida que giramos su eje (un potenciómetro es una resistencia variable), mientras que con el *paddle* utilizamos uno lineal, lo que quiere decir que a igual cantidad de grados de giro obtenemos la misma variación en la resistencia que presenta el potenciómetro a la corriente eléctrica. Por ser continuo el movimiento del eje, desde el principio al fin (unos 270°, aproximadamente) el tipo de señal que nos produce viene también cuantificada linealmente, aunque el ordenador la convierte en valores escalonados por la naturaleza digital del sistema, pero estos valores varían de uno en uno, por lo que la sensación de continuidad está asegurada.

Para disipar dudas continuamos diciendo que ambos tipos de accesorio pueden ser utilizados tanto con el **Vic 20** como con el **C-64.** La conexión es directa por medio de los *ports* que el fabricante ha dispuesto en el exterior de los ordenadores. El tipo de conector es exactamente el mismo para ambos, respondiendo al están-

PATILLA	VIC 20	CBM-64	
	PORT	PORT 1	PORT 2
1	JOY 0	JOY A0	JOY B0
2	JOY 1	JOY A1	JOY B1
3	JOY 2	JOY A2	JOY B2
4	JOY 3	JOY A3	JOY B3
5	POT Y	POT AY	POT BY
6	LAPIZ OPTICO	BOTON A/L.O.	BOTON B
7	+5V	+5V	+5V
8	MASA	MASA	MASA
9	POT X	POT AX	POT BX

dar creado en su día por **Atari**, por ello también es posible utilizar los modelos de esta marca.

Exteriormente, el aspecto de los conectores de ambos ordenadores es idéntico. La principal diferencia consiste en que el **Vic 20** dispone de uno solo, mientras que el **C-64** lleva incorporados dos. La apariencia de estos *ports* puede verse en la figura 3, junto con la tabla que muestra la correspondencia entre terminales del conector. Este conector corresponde a un estándar llamado **tipo D**, en este caso es de 9 patillas.

Aun cuando tanto el **Vic 20** como el **C-64** pueden utilizar los mismos modelos de *joystick* y *paddles*, la manera en que estos accesorios para juegos se comunican con la circuitería es diferente, por haber sido diseñados en épocas próximas, pero distintas, utilizándose en cada momento lo que el mercado de los *chips* ofrecía. El **Vic 20** utiliza un *chip* denominado VIA 6522 que corresponde a las siglas de *Versatile Interface Adapter*, mientras que el **C-64** utiliza otro más sofisticado, el CIA *(Complex Interface Adapter)* 6526. En realidad ambos ordenadores utilizan un par de este tipo de *chips* cada uno.

Comenzamos con el **Vic 20**. Los dos VIA disponen de una serie de registros internos, pero de momento nos centraremos en el conocido por DDR *(Data Direction Registers)* o Registros de Dirección de los Datos. El VIA dispone de dos *ports* de Entrada/Salida y cada uno de los dos DDR (A y B) se encarga de definir qué patilla de los *ports* es de entrada o de salida de datos; es decir, si uno determinado envía datos al exterior, procedentes del ordenador, o si, por el contrario, los toma desde fuera para alimentar de información al ordenador, como es el caso de la utilización del *joystick*. Recapitulemos, cada bit del byte de cada DDR controla que una patilla de un determinado *port* sea de entrada o de salida de datos. A su vez nosotros depositamos, mediante programación, el byte adecuado para cada DDR. Si un bit es puesto a 1, su patilla correspondiente será de salida. En caso contrario es de entrada.

Por peculiares razones de diseño sucede que uno de los VIA se hace cargo de tres de los pulsadores y el botón de fuego del *joystick* y el otro VIA controla al cuarto. Esto no nos afecta en otra cosa más que en lo engorrosa que se hace la programación. Pero con la ayuda del útil POKE y conociendo las direcciones de memoria adecuadas, obtendremos unos resultados óptimos. Lo que acabamos de ver se materializa de la siguiente forma. Las direcciones de memoria asociadas a cada VIA con-

trolan lo siguiente: 37137 el Norte, Oeste, Sur, y botón de disparo de los pulsadores dispuestos en forma de brújula. El pulsador que corresponde a la posición del Este va asociado con la 37152. La utilización de PEEK para conocer el contenido de los registros situados en estas direcciones de memoria nos permiten saber en qué dirección se ha movido el *joystick* o si hemos hecho fuego. Sin embargo, los primero que debemos hacer es poner en orden los DDR, cuyas direcciones de memoria son 37139 y 37154. La primera es el DDR A y la segunda, el B. Para configurarlos correctamente debemos recurrir a:

```
1000 POKE 37139 , 0
1010 POKE 37154, 127
```

Siendo ficticios los números de línea de programa. Después de esto se puede utilizar PEEK, pero con reservas, todavía hay que desmenuzar su información para ver cómo la hacemos útil a nuestros fines. La información contenida en 37137 es la siguiente: el bit 2 nos dice que hemos pulsado el Norte; el bit 3, el Sur; el bit 4, el Oeste, y finalmente, el 5 denuncia el fuego. Por su lado, 37152 indica en su bit 7 que ha sido presionado el pulsador del Este.

En la sección destinada al montaje práctico de un *joystick* proporcionamos una interesante rutina de utilización, acompañada de varios ejemplos.

Analicemos ahora cómo utiliza el **C-64** su par de CIAs para leer los *joysticks*. Igual que con el VIA aquí disponemos de dos *ports* paralelo de 8 bits (un byte) de longitud de palabra, aunque también hay dos temporizadores de intervalo, un reloj diario y un *port* serie de Entrada/Salida sincrono. Una de las CIAs queda destinada al *port* de usuario y al conector serie, capaz de comunicarse con diversos periféricos tales como la impresora, la unidad de *diskettres*, etc.

La otra CIA es la que nos interesa, actúa como *interface* de conexión con el teclado, lectura del *cassette* y lectura de *joysticks* y lápiz óptico.

Igual que hicimos con el **Vic 20** ahora seguiremos hablando de las direcciones de memoria del **C-64** que afectan a los *ports* de control. El *port* 1 está conectado a la dirección 56321 y el *port* 2, a la 56320. Solamente se utiliza la mitad del byte de cada una de ellas, los cuatro bits de menor peso. Mediante la operación lógica AND se obtiene la información:

A = PEEK (56321) AND 15
B = PEEK (56320) AND 15

Igual que en el caso anterior, la plena utilización de estas direcciones con fines prácticos serán desglosadas en el montaje y en la segunda parte de este artículo.

Entremos a ver lo que ocurre con los *paddles*. Pero aclaremos previamente que es necesaria la existencia de alguna circuitería que se encargue de la conversión A/D (analógico/digital), que transforma las señales analógicas (resistencia del potenciómetro) en valores digitalizados, como aclarábamos hace unos párrafos. En ambos casos, **Commodore** ha confiado esta tarea a *chip* específicos de la marca. En el caso del **Vic 20** es el *chip* SID 6560 quien lo hace, por medio de las patillas **POT X** y **POT Y** (5 y 7 del conector, respectivamente). El **C-64** utiliza el *chip* **SID 6581** que, recordemos, es el sintetizador de sonido. Leer los *paddles* es una operación más o menos parecida a la lectura de los *joysticks*. Aquí no hay que comprobar el estado de cuatro pulsadores, sino los valores digitales correspondientes a la cantidad de giro efectuada en el *paddle* y cuyo valor puede oscilar entre 0 y 255. Por lo que a los botones de disparo se refiere, se utilizan las mismas direcciones de memoria y la misma patilla del conector que se usó con los *joysticks*.

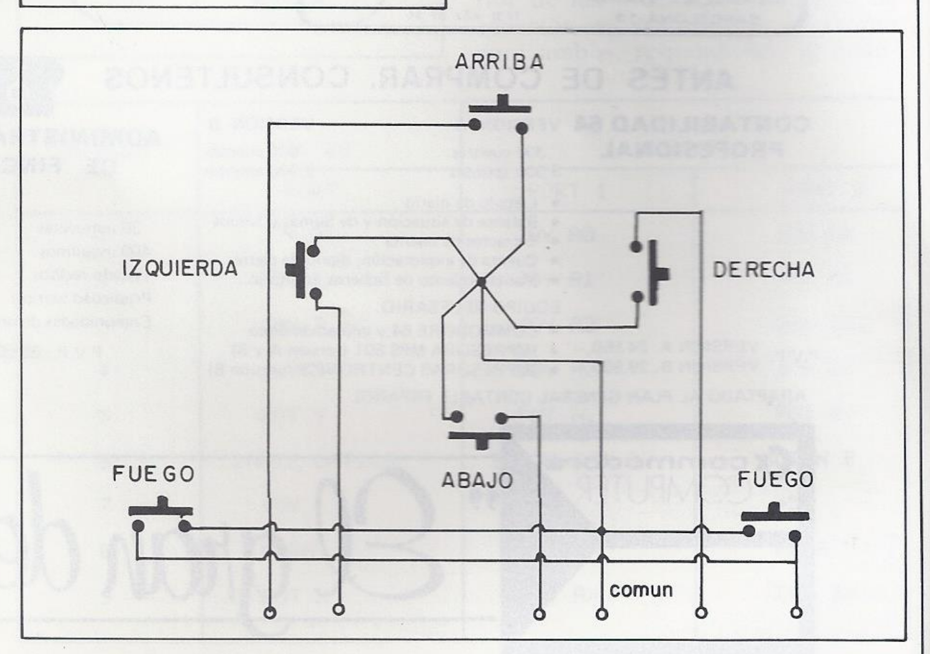
En las direcciones 36872 y 36873, respectivamente, se almacenan los valores del *paddle* X e Y. Por tanto:

```
1000 X = PEEK (36872)
1010 Y = PEEK (36873)
```

proporcionan el valor correspondiente a cada uno de ellos. Las direcciones equivalentes para el **C-64** son 54297 y 54298 para **POT X** y **POT Y**, respectivamente, pero no olvidemos que hay dos *ports* de control en este caso y la posibilidad de conectar cuatro *paddles*, dos por cada *port*. Como veremos, el misterio está en la dirección 56320.

(Continuará)

ARRIBA
POT. Y
ABAJO
IZQUIERDA
POT. X
DERECHA



PARA LLEGAR MAS LEJOS...

SPECTRAVIDEO™



MKII







MICROSOFT BASIC ★ FORTRAN 80 ★ Microsoft Logo ★ **COLECO cartuchos**
CP/M ★ Cobol 80 ★ MACROENSAMBLADOR Z80 ★ **Ensamblador CP/M**

- **MSX.** Es el sistema escogido por 16 fabricantes japoneses de ordenadores (SONY, PANASONIC, PIONER, SANYO, SPECTRAVIDEO, etc.) siendo compatibles todos los programas creados por cada uno de los fabricantes.

CARACTERISTICAS GENERALES	SPECTRAVIDEO 328	SPECTRAVIDEO 318	APPLE II e	COMMODORE 64	ATARI 600 xL	BBC MODELO B	DRAGON 32	ORIC ATMOS
PROCESADOR	Z80 A	Z80 A	6502	6510	6502	6502	6809 E	6502
RELOJ EN MHz	4 MHz	4 MHz	1 MHz	1 MHz	1.8 MHz	2 MHz	1 MHz	1 MHz
MEMORIA STANDARD EN RAM	80 K	32 K	64 K	64 K	16 K	32 K	32 K	48 K
AMPLIABLE HASTA	144 K	144 K	–	–	64 K	–	64 K	64 K
MEMORIA STANDARD EN ROM	32 K	32 K	16 K	20 K	24 K	16 K	16 K	16 K
AMPLIABLE HASTA	96 K	96 K	–	–	–	–	–	–
MICROSOFT BASIC EXTENDIDO	SI	SI	–	–	–	NO	SI	SI
NUMERO DE TECLAS	87	51	62	66	57	73	53	57
TECLADO CON GRAFICOS PREDEFINIDOS	SI	SI	–	SI	SI	SI	NO	NO
TECLAS DE FUNCION PROGRAMABLES	10	10	–	8	4	10	NO	NO
MANDO JOYSTICK INCORPORADO	NO	SI	–	NO	NO	NO	NO	NO
COLORES	16	16	16	16	16	16	9	8
SPRITES	32	32	–	8	8	–	?	–
RESOLUCION (Puntos de pantalla)	256×192	256×192	280×192	320×200	320×192	256×640	256×192	240×200
TEXTO EN PANTALLA	40×24	40×24	40×24	40×25	40×24	40×32	32×16	40×28
CANALES DE SONIDO	3	3	1	3	4	1	3	3
OCTAVAS POR CANAL	8	8	4	9	?	3	5	8
A.D.S.R. ENVOLVENTE	SI	SI	NO	SI	NO	SI	NO	SI
CAPACIDAD UNIDAD DE DISCO	256 K	256 K	140 K	170 K	127 K	100 K	?	?
CENTRONICS	SI (opcional)	SI (opcional)	SI (opcional)	–	SI (opcional)	SI (opcional)	SI	SI
RS232	SI (opcional)	SI (opcional)	SI (opcional)	–	SI (opcional)	SI (opcional)	–	–
80 COLUMNAS	SI (opcional)	SI (opcional)	SI (opcional)	SI (opcional)	SI (opcional)	–	–	–
PRECIO P.V.P.	**76.000**	**49.900**	**166.618**	**79.900**	**58.500**	**140.000**	**67.800**	**59.500**



HARDWARE
DISTRIBUIDOR OFICIAL DE SPECTRAVIDEO INTERNATIONAL LTD
PASEO DE LA CASTELLANA, 179 - MADRID-16
TELEFONO (91) 279 31 05

DELEGACION EN CATALUÑA: ACE, S.A.
Tarragona, 100 - Barcelona-15.
Teléfono (93) 325 10 58

YA DISPONIBLE EN:



... Y EN TODAS LAS TIENDAS ESPECIALIZADAS.

Construye tu JOYSTICK

Indudablemente, la mecánica constituye la parte más delicada de la fabricación de un *joystick*. El diseño de las piezas que conforman la palanca de control, que activa los pulsadores, representa una fase crítica. Se trata de un producto de alto nivel de consumo, que debe conjugar lo atractivo de su diseño externo con la comodidad de manejo. En realidad lo que diferencia a un modelo de otro es la apariencia externa, terreno en el que compiten las diversas marcas entre sí.

Como se hace evidente, la posibilidad de desarrollar y construir piezas mecánicas de plástico no es algo que esté al alcance de cualquier aficionado, aparte de la dudosa rentabilidad económica que esto representaría, sale más barato comprar un *joystick* en la tienda.

Sin embargo, existen alternativas a la mano de cualquiera con ganas. El precio es muy inferior, sacrificándose la comodidad de manejo, pero elevando a su cota máxima el entusiasmo por la obra propia. Es el hágaselo-usted-mismo en su más elevado grado.

Simplemente con cinco o seis pulsadores cuyo circuito esté normalmente abierto (es decir, que no haya continuidad entre sus bornes mientras no sea presionado). Luego habrá que disponerlos adecuadamente en una pequeña caja de plástico, soldar unos cuantos cables y poner el conector tipo D, de 9 terminales en su extremo. En cuanto a este último, cabe decir, que se puede utilizar

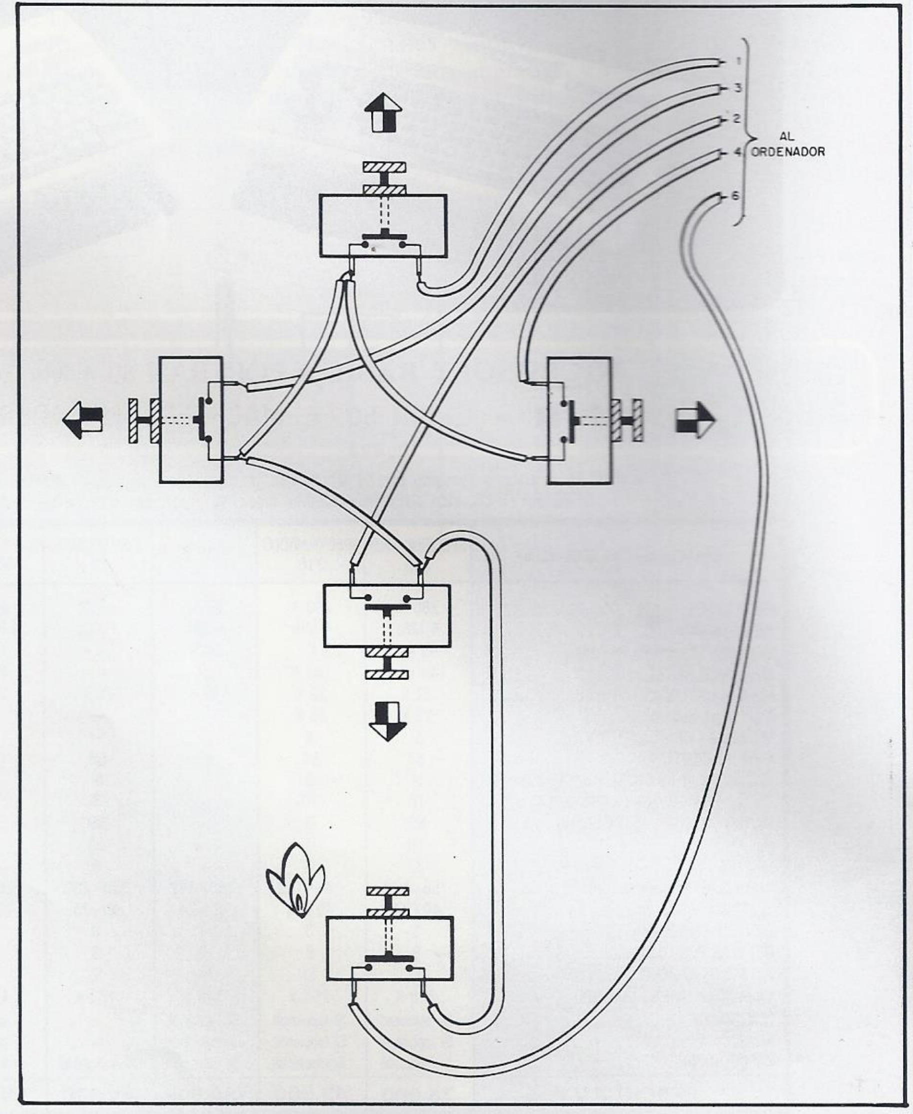


Figura 1. Diagrama esquemático de la conexión de los pulsadores del *joystick*. En el interior de cada uno se ha dibujado el símbolo del circuito que incorporan.

exactamente el mismo tanto para el Vic-20 como para el C-64.

Construye tu JOYSTICK

En la figura 1 se muestra un diagrama esquemático de la constitución del montaje. Aparecen, por un lado, cuatro pulsadores formando una cruz. Cada uno de ellos servirá para controlar el movimiento del cursor, marcianito, etc., en uno de los cuatro sentidos, o un movimiento en diagonal cuando se pulsen dos simultáneamente. Claro está que en cualquier caso el programa que utilicemos deberá incorporar una rutina, o serie de ellas, que posibiliten un reflejo del movimiento en la pantalla. Más adelante en este artículo mostraremos algunos ejemplos que se pueden incluir en los programas que nosotros mismos desarrollemos.

El quinto pulsador será el de fuego, llamado así porque normalmente se utiliza para disparar cuando estamos corriendo un programa de juegos. Aunque en la figura sólo aparece uno, es factible disponer otro (u otros) en

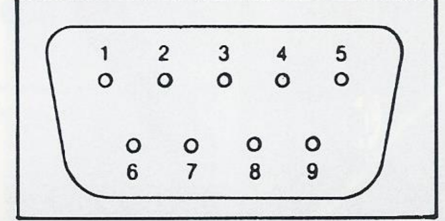


Figura 2. Aspecto que ofrecen los *ports* de control, vistos desde el frente.

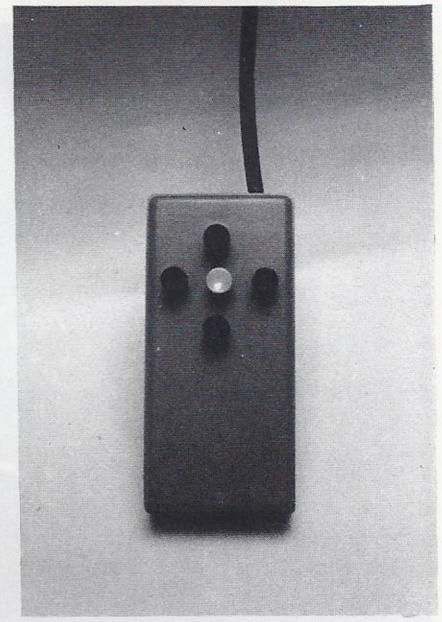


Figura 3. *Joystick* experimental.

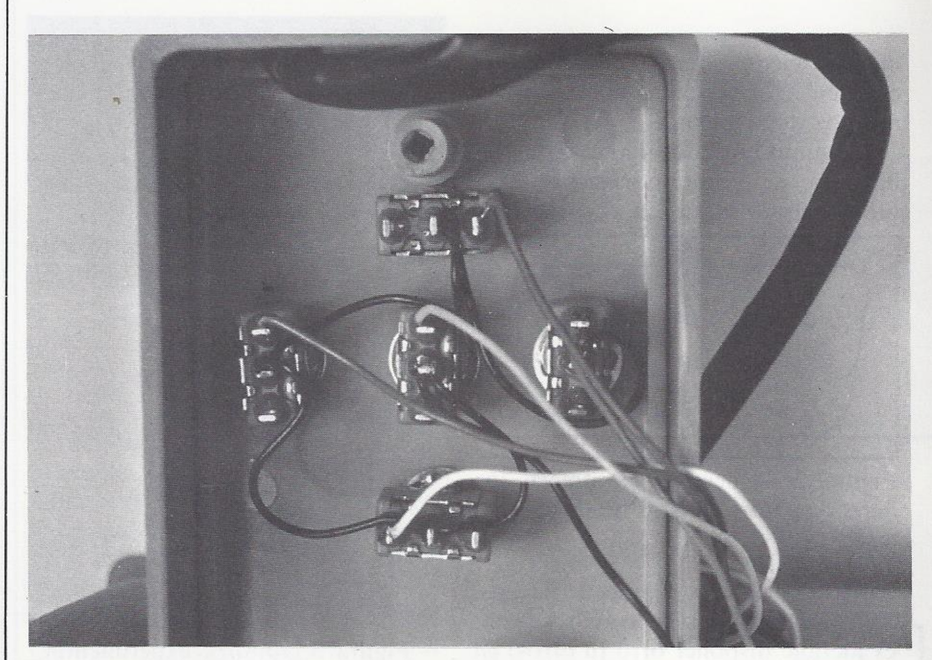


Figura 4. Vista del interior del *joystick*.

distintas partes de la caja, para mayor comodidad del disparo. En tal caso deberá conectarse el segundo pulsador en modo paralelo con el primero. Esto quiere decir que uno de los hilos que proceden del pulsador va a una borna del primer pulsador de fuego y el otro hilo al que ha quedado libre. La conexión al ordenador no sufre alteraciones. Los números que aparecen en el extremo de los hilos indican el número de patillas a que deben ser conectados en el *port* de control del ordenador, siempre por medio del conector tipo D. En el caso del **C-64** existen dos *ports* de control (el 1 y el 2), pero el diagrama es exactamente el mismo. En la figura 2 se ha dibujado una vista del *port*, en la disposición en que puede verse desde el exterior del ordenador. Conviene ser cuidadosos al soldar los hilos a los terminales del conector que ha de enchufarse al *port*, pues recordemos que tiene lugar un efecto similar al de un espejo si lo observamos desde la parte frontal. Lo más sencillo es hacer que el diagrama de la figura 2 coincida con el lado que contiene los terminales destinados a la soldadura.

En la figura 3 se puede ver el *joystick* que nosotros hemos elaborado. Desde luego su aspecto difiere mucho de uno comercial pero funciona perfectamente. Por motivos de claridad, se dispuso un solo botón de fuego en el centro de la cruz que forman los de dirección, aunque no es el lugar más cómodo. En cuanto a la elección del tipo de pulsadores no existe una regla determinada, sirve cualquiera cuyo circuito esté normalmente abierto, como decíamos antes, hecho este de gran importancia. Siempre es factible diseñar algún otro tipo de dispositivo de más vistoso montaje mecánico. Cualquier lector un poco "manitas" enseguida pensará en cómo mejorar este diseño. En tal caso que no dude en hacérnoslo conocer para divulgar su idea en la revista.

La foto de la figura 4 ofrece una visión general de cual sería el aspecto que ofrecen las soldaduras de los hilos a los diversos pulsadores.

Las figuras 5 y 6 representan las áreas parciales del esquema teórico tanto de **Vic 20** como **C-64** (en la segunda), cuya única misión consiste en conocer qué pasa al otro lado del *port* de control, pero desde luego carece de otros fines prácticos en el montaje que nos ocupa.

Construye tu JOYSTICK

Una vez montado el *joystick* enseguida querremos ver como funciona. Comencemos por el **Vic 20**. Después de haber leído la primera parte del artículo sobre *joysticks*, sabemos que hay que configurar los registros de dirección de datos, colocando información en las direcciones de memoria 37139 y 37154. La primera línea del programa hace esto. La información es depositada en las variables A y B, que la toman a su vez de las direc-

```
10 A=PEEK(56321) AND 15:REM**LEE EL 1
20 B=PEEK(56320) AND 15:REM**LEE EL 2
30 PRINT A;"     ";B
40 F1=-((PEEK(56321) AND 16)=0):REM**FUEGO 1
50 F2=-((PEEK(56320) AND 16)=0):REM**FUEGO 2
60 PRINT F1;"   ";F2
70 FORX=1 TO 50:NEXTX
80 GOTO 10
```

```
 1=Norte
 2=Sur
 4=Oeste
 8=Este
 5=Noroeste
 6=Suroeste
 9=Noreste
10=Sureste
```

Figura 5. Vista parcial del **Vic 20**. El esquema teórico muestra donde terminan las conexiones del *port* de control.

```
10 POKE 37139,0:POKE 37154,127
20 A=PEEK(37137)
30 B=PEEK(37152)
40 PRINT A;"        ";B
50 FOR X=1 TO 1000:NEXTX
60 GOTO 20
```

Figura 6. Vista parcial del esquema teórico del **C-64**.

ciones destinadas al *port* de control. La línea 40 realiza una operación OR lógica con ambos bytes de información y en las líneas 50 a 90 se comprueba que botón ha sido activado, por medio de "máscaras" y la operación lógica AND. Obsérvese que cada uno de los números que aparecen entre paréntesis es una potencia de 2 y, por tanto, su equivalente en base 2 corresponde a uno solamente, dispuesto en diferente lugar del byte.

Una vez establecido si ha sido o no pulsado un botón el programa salta a una determinada dirección de memoria, donde se halla la parte que indica al ordenador, lo que debe hacer en tal supuesto. Como es lógico los números 100, 200, 300, 400 y 500 que aparecen en estas líneas son supuestos, debiendo alterarse en función del programa que desarrollemos. La utilización en el **C-64** es más sencilla. La información se deposita directamente

Construye tu JOYSTICK

desde el *joystick* en las direcciones 56320, para el *port 2*, y 56321, para el *port 2*. Lo único que resta por hacer es leer el que nos interese en el momento adecuado para el programa. En la figura 8 aparecen las sentencias adecuadas, así como los valores proporcionados por la acción de cualquiera de los *joystick*.

Concurso

Memory

CBM 64

```
30 GOTO3000
50 PRINT"[illegible] NIVEL DIFICULTAD"
52 PRINT"[illegible]"
54 PRINT"[illegible]"
56 PRINT"[illegible]"
58 PRINT"[illegible]DIFICIL                FACIL"
60 PRINT"[illegible]          PULSE   <F>  O  <D>  "
61 PRINT "      PULSE <ESPACIO> PARA JUGAR"
62 PRINT"IIIIIIIIIIII]"
64 PRINT"[illegible]    ";
69 ND=1
70 GET ND$
72 IF ND$="D"AND ND>1 THEN ND=ND-1:GOTO 80
74 IF ND$="F"AND ND<9 THEN ND=ND+1:GOTO 90
75 IFND$=" "THEN GOTO 95
76 GOTO 70
80 PRINT"[illegible]";
84 GOTO 70
90 PRINT"   ";
94 GOTO 70
95 ND=(ND-1)*100
96 PRINT"[illegible]PREPARESE!!"
97 PRINT"[illegible]VAN A SALIR"
98 PRINT"[illegible]LAS LETRAS"
99 PRINT"[illegible]!!! MEMORICELAS !!!";
100 DIM X$(8,6)
105 POKE 53281,9:POKE53280,9
110 C=0:CA=0:CF=0:XP=1794
120 FOR J=1 TO 6
130 FOR I=1 TO 8
140 LET X$(I,J)="Z"
150 NEXT I:NEXT J
200 J=INT(RND(1)*6+1)
210 I=INT(RND(1)*8+1)
220 IF X$(I,J)<>"Z" THEN GOTO 200
230 READ X$
240 LET X$(I,J)=X$
260 J=INT(RND(1)*6+1)
270 I=INT(RND(1)*8+1)
280 IF X$(I,J)<>"Z" THEN GOTO 260
290 LET X$(I,J)=X$
300 C=C+1
310 IF C<24 THEN GOTO 200
320 PRINT"[illegible]";
340 FORJ=1 TO 6
350 FOR I=1 TO 8
360 PRINT"[illegible]";X$(I,J);
365 TT=50 :GOSUB2200
370 FORC=0 TO ND :NEXT C
380 PRINT"[illegible] ";
390 NEXT I
400 PRINT"[illegible]";
410 NEXT J
```

Memory es un programa para el *Commodore 64*, escrito por **Francisco Roig Serra** y que llega a nuestra revista desde **Barcelona.** Se trata de un juego que consiste en lo siguiente: en la pantalla van apareciendo, uno tras otro, **48 caracteres,** cada uno de los cuales aparece repetido. A medida que van apareciendo son ocultados por un pequeño rectángulo. El juego consiste en memorizar la posición de los caracteres para posteriormente agruparlos en parejas, con el mínimo

PREMIADO CON 5.000 PESETAS



número de fallos posibles. Para formar las parejas hay que mover el *cursor* mediante las teclas M, N y las teclas de cursor CRSR. Si los dos caracteres destapados coinciden, entonces desaparecen de la pantalla mientras se escucha un agradable sonido de felicitación.

El juego acaba cuando se han formado **24** parejas de caracteres, apareciendo la calificación final en **tanto por ciento** de aciertos y el número de fallos cometidos.

```
420 PRINT"■■▯TTTK▮D";
425 LET J=6:LET I=1:LETTT=1000
430 GET A$
440 IFA$="."ANDI<8THENI=I+1:XP=XP+3:PRINT"■■■ ■ <▮D";:IFK>1THENX=X+1:GOTO450
445 IFA$="."THENGOTO430
450 IFA$=","ANDI>1THENI=I-1:XP=XP-3:PRINT"■ ■■■ ■■■■K▮D";:IFK>1THENX=X-1:GOTO460
455 IFA$=","THENGOTO430
459 IFA$="N"ANDJ>1THENXP=XP-40*3
460 IFA$="N"ANDJ>1THENJ=J-1:PRINT"■■■ ■ ■■▯TTTK▮D";:IFK>1THENY=Y-1:GOTO469
465 IFA$="N"THENGOTO430
469 IFA$="M"ANDJ<6THENXP=XP+40*3
470 IFA$="M"ANDJ<6THENJ=J+1:PRINT"■■■ ■ ■■QQQK▮D";:IFK>1THENY=Y+1:GOTO480
475 IFA$="M"THENGOTO430
480 IFA$=" "ANDK>1ANDPEEK(XP)=160THEN PRINT"■■";X$(I,J);"■";:GOTO 900
490 IFA$=" "ANDK<2ANDPEEK(XP)=160THENPRINT"■■";X$(I,J);"■";:X$=X$(I,J):K=2
500 GOTO 430
900 FOR L=0 TO 100 :NEXT L
910 IF X$=X$(I,J)THENGOSUB 2000: PRINT"■■  ■■■ ";:CA=CA+1:GOTO930
920 IF X$<>X$(I,J)THENGOSUB2200:PRINT"■■■ ■■ ■■■ ";:CF=CF+1
930 FOR L=0TOABS(X-1)
940 IFX>0THENPRINT"■■■";:XP=XP-3
950 NEXTL
955 FOR L=0TOABS(X+1)
960 IFX<0THENPRINT"■■■";:XP=XP+3
965 NEXTL
970 FORL=0TOABS(Y-1)
980 IFY>0THENPRINT"TTT";:XP=XP-40*3
990 NEXT L
995 FORL=0TOABS(Y+1)
1000 IFY<0THENPRINT"QQQ";:XP=XP+40*3
1005 NEXTL
1010 IF X$=X$(I,J)THEN PRINT"■ ■■";
1020 IF X$<>X$(I,J)THEN PRINT"■■ ■■";
1025 LETI=I-X:LETJ=J-Y
1027 PRINT"■■■■K▮D";
1030 X=0:Y=0:K=1
1035 IF CA=24 THEN GOTO 1500
1040 GOTO500
1500 PRINT"■■■■■QQQQQQ■ CALIFICACION ■"
1510 PRINT:PRINT
```

Viene de la página anterior

```
1520 PRINT      "■■■■■■■FALLOS";"■......■";CF
1525 PRINT:PRINT:PRINT:PRINT"■■■■■■■% DE ACIERTOS:";INT((24/(24+CF))*100);" %"
1530 PRINT:PRINT:PRINT:PRINT:PRINT:PRINT"■■■■■■■■■■■■*****************"
1540 PRINT:PRINT:PRINT:PRINT"■■ PARA OTRO JUEGO PULSE O "
1541 POKE54296,15:POKE54295,0:RESTORE
1542 POKE54278,0:POKE54277,64+13:POKE54275,99:PP=2500
1543 GOSUB4000
1545 JJ=JJ+1:IFJJ=40 THENGOTO1549
1547 GOTO 1543
1549 POKE54296,0
1550 GET O$:IF O$<>"O"THEN GOTO 1550
1555 POKE 53281,6
1556 POKE54296,15:POKE54295,0:RESTORE
1557 GOSUB4000
1558 JJ=JJ+1:IFJJ=40 THENGOTO1560
1560 RUN
2000 VV=54272
2005 POKE54295,0:POKE54296,15
2010 POKEVV+6,0:POKEVV+5,12:POKEVV+1,200
2020 POKEVV+3,40
2030 FORNN=1TO4
2040 POKEVV+4,0:POKEVV+4,65
2050 FORMM=1TO500:NEXTMM
2060 NEXTNN
2070 POKEVV+4,0
2075 POKE54296,0
2080 RETURN
2200 POKE54295,0:POKE54296,15
2210 VV=54272:POKEVV+6,0:POKEVV+5,12
2220 POKEVV+1,20:POKEVV+4,129
2230 FORNN=1TOTT  :NEXTNN
2240 POKEVV+4,0
2250 POKE54296,0
2260 RETURN
3000 PRINT"■":A=0
3002 PRINT"■■■■■■■■■■■■■■■■■■■■■■■■■■A GAME FOR"
3004 PRINT"■■■■■■■■■■■■■■■■■CBM-64"
3006 FORI=0 TO 2000:NEXTI
3008 PRINT"■"
3010 POKE53280,9:POKE53281,9
3011 POKE54296,15:POKE54295,0
3012 VV=54272
3013 POKEVV+6,0:POKEVV+5,64+13
3014 POKEVV+3,99:PP=2500
3020 PRINT"■■■    ●  ●●●●  ●    ●     ●    ●●   ●   ●"
3021 PRINT"■■●  ●●  ●      ●● ●●    ●  ●    ●  ●  ●  ●"
3022 PRINT"■■●  ● ●  ●●●   ● ● ●    ●  ●    ●  ●    "
3023 PRINT"■■●    ●  ●      ●   ●   ●  ●    ●●    ●"
3024 PRINT"■■●    ●  ●      ●   ●   ●  ●    ●  ●  ●"
3025 PRINT"■■●    ●  ●●●●   ●   ●     ●     ●  ●  ●"
3026 GOSUB 4000
3028 IFA=0THEN FORI=0TO3000:NEXTI
3030 FOR I=0 TO 100:NEXTI
3031 IF A=7 THEN A=0:GOTO3050
3032 PRINT"■"
3033 A=A+1
```

```
3034 FORI=0 TO A
3036 PRINT""
3038 NEXTI
3040 GOTO 3020
3050 FORJ=0 TO 15
3060 FORI=55976 TO 56215
3070 POKEI,J
3080 NEXTI
3085 IF J=3 THEN PRINT"A GAME FOR"
3086 IF J=5 THEN PRINT"COMMODORE CBM 64"
3087 IF J=9 THEN PRINT" COPYRIGHT FRANCISCO ROIG. 1984."
3088 IF J=12 THEN PRINT"TODOS LOS DERECHOS RESERVADOS."
3090 NEXTJ
3091 GOSUB4000
3092 GOTO3091
3095 POKE54296,0
3100 PRINT"NECESITAS INSTRUCCIONES?"
3110 GET S$:IF S$="S"THEN GOTO 3120
3112 IF S$=""THEN GOTO 3110
3114 GOTO50
3120 PRINT""
3130 PRINT"EN LA PANTALLA APARECERAN 24 PARES"
3140 PRINT"DE CARACTERES OBSERVABLES SOLO POR UN"
3150 PRINT"PEQUENO LAPSO DE TIEMPO."
3160 PRINT"INTENTE MEMORIZAR LA POSICION DE"
3170 PRINT"CADA CARACTER, PUES EL JUEGO CONSISTE"
3180 PRINT"EN DESTAPAR UNA POSICION CUALQUIERA Y"
3190 PRINT"SEGUIDAMENTE LA POSICION QUE CONTIENE"
3200 PRINT"EL MISMO CARACTER."
3210 PRINT"A MEDIDA QUE USTED VA ENCONTRANDO"
3220 PRINT"PARES DE CARACTERES,ESTOS DESAPARECEN."
3230 PRINT"EL JUEGO TERMINA CUANDO SE HAN"
3240 PRINT"HALLADO LOS 24 PARES DE CARACTERES."
3250 PRINT"LOS FALLOS SON CONTABILIZADOS."
3260 PRINT" PARA CONTINUAR PULSE ESPACIO "
3270 GET S$:IFS$<>" "THEN GOTO 3270
3280 PRINT""
3290 PRINT"PARA MOVER EL CURSOR:"
3300 PRINT""
3310 PRINT"NORTE........"
3320 PRINT"SUR.........."
3330 PRINT"ESTE........."
3340 PRINT"OESTE........"
3345 PRINT"PARA DESTAPAR PULSE <ESPACIO>"
3350 PRINT" PARA CONTINUAR PULSE ESPACIO "
3360 GETS$:IF S$<>" "THEN GOTO 3360
3370 GOTO50
4000 READWW:IFWW=0THEN GOTO3095
4010 RR=INT(WW/10):QQ=(WW-10*RR)*PP
4020 POKE54273,QQ/256:POKE54272,QQAND255
4030 POKE54276,0:POKE54276,65:FORNN=1TO70*(RR+1):NEXTNN
4040 RETURN
9990 DATA 25,6,24,5,3,3,3,4,5,6,25,6,24,5,3,3,3,64,3,3,5,4,4,5,3,3,5,4,5,6
9995 DATA 15,6,24,5,3,3,3,4,5,6,25,6,24,5,3,3,3,94,0
10000 DATA A,B,C,7,E,F,G,H,I,5,K,L,M,4,O,P,3,R,S,T,U,2,X,Y
```

PROGRAMA commodore COMPROBADO

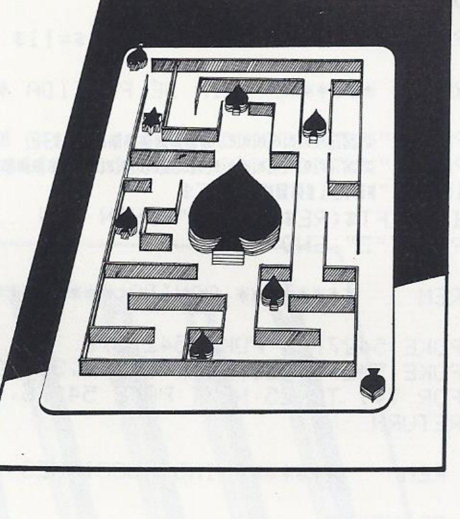
Concurso

laberinto de picas

CBM 64

Para participar en el concurso mensual de nuestra revista, Lorenzo López Martos nos envía, desde Barcelona, este juego para el C-64 titulado **LABERINTO DE PICAS.** El juego consiste en desplazarse por un intrincado laberinto "comiéndose" todas las picas que aparecen desperdigadas en él. Las picas son uno de los cuatro símbolos de la baraja francesa (rombos, corazones, tréboles y picas). Para limpiar el laberinto se dispone de un tiempo limitado, que puede ser de dos minutos, si se elige el nivel de dificultad fácil, o sólo de uno en el nivel difícil. Cada vez que se acaba con todas las picas de un laberinto, dentro del tiempo disponible, hay que prepararse para empezar de nuevo, ya que aparecerá una nueva pantalla con dos picas más que en la anterior. Cuando se completa una pantalla antes de transcurrido el tiempo, los segundos sobrantes se acumulan para poderlos emplear en la siguiente pantalla. El movimiento por el laberinto se controla desde el teclado, mediante cuatro teclas para las cuatro direcciones posibles de movimiento. En las instrucciones incluidas en el programa, que se pueden leer nada más escribir *RUN*, se indica cuáles son estas teclas entre las explicaciones generales del juego. Enhorabuena al autor de este programa que esperamos sea del agrado de todos nuestros lectores. Ah, se nos olv.daba, el jugador viene representado en el laberinto por un sencillo asterisco.

```
1 REM ************************************
2 REM *                                  *
3 REM *      LABERINTO DE PICAS          *
4 REM *      MEMORIA:3.7 KBYTES          *
5 REM *      (C)   SERGIO LOPEZ          *
6 REM *                                  *
7 REM ************************************ :
8 POKE 54296,15:GOSUB 10000:GOSUB 20000
9 GOTO 20
10 REM ***** SUBRUTINA DEL TIEMPO *****
11 TP$=STR$(TA-VAL(TI$))
12 PRINT"▓▓   TIEMPO: ";TP$;"      ";
13 IF VAL(TP$)<=0   THEN GOTO 1000
15 PRINT"▓▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶PUNTOS: ";AA:RETURN
20 DEF FN PO(A)=A+JX+JY*40
30 POKE 198,0
40 AA=0:S=1024
50 POKE 53280,6:POKE 53281,6:PRINT"▓"
99 :
100 REM ******* CREAR LABERINTO ******
101 :
105 DIM A(3)
110 A(0)=2:A(1)=-80:A(2)=-2:A(3)=80
120 A=S+541
130 PRINT"▓";
140 FOR JY=0 TO 24
153 FOR JX=0 TO 38
155 POKE FN PO(1024),160
157 NEXT JX
160 NEXT JY
210 POKE A,4
220 J=INT(RND(1)*4):X=J
230 B=A+A(J):IF PEEK(B)=160 THEN POKE B,J:POKE A+A(J)/2,32:A=B:GOTO 220
240 J=(J+1)*-(J<3):IF J<>X THEN 230
250 J=PEEK(A):POKE A,32:IF J<4 THEN A=A-A(J):GOTO 220
299 :
300 REM ** DIRECCIONES DE MOVIMIENTO **
```



```
301 :
302 A(0)=1:A(1)=40:A(2)=-1:A(3)=-40
303 REM ****  POSICIONA LAS PICAS *****
304 NP=10:PICAS=NP
305 FOR I=1 TO NP
310 JY=INT(RND(0)*25)
320 JX=INT(RND(0)*38)
330 IF PEEK(FN PO(S))=32 THEN POKE (FN PO(S)),65:GOTO 345
340 GOTO 310
345 NEXT
346 REM *INICIALIZA EL TIEMPO DE JUEGO*
347 IF NP>10 THEN TA=TA+DI:GOTO 350
348 TA=159
349 REM **** POSICIONA AL JUGADOR ****
350 IF NP>10 THEN PO=POX:GOTO 405
360 JY=INT(RND(0)*24)
370 JX=INT(RND(0)*38)
390 IF PEEK(FN PO(S))=32 THEN POKE (FN PO(S)),42:PO=FN PO(S):GOTO 405
400 GOTO 360
401 :
402 REM **** INICIO DEL JUEGO ********
403 :
405 PRINT"██   PULSE UNA TECLA PARA EMPEZAR            █":POKE 198,0
406 GET A$:IF A$="" THEN 406
407 PRINT"██                                            █":PP=0
408 IF NP=10 THEN TI$="000000":GOTO 410
409 TI$=TP$
410 IF PICAS=0 THEN 500
411 :
412 REM ******** MOVIMIENTO **********
413 :
414 GET A$:GOSUB 10:IF PEEK(197)=0 THEN 460
415 IF A$=":" THEN PP=0
420 IF A$="A" THEN PP=1
430 IF A$="L" THEN PP=2
440 IF A$="Q" THEN PP=3
460 PO=PO+A(PP)
465 IF PEEK(PO)=65 THEN AA=AA+100:PICAS=PICAS-1:GOSUB 10:GOSUB 9000
470 IF PEEK(PO)=32 OR PEEK(PO)=65 THEN POKE PO,42:POKE PO-A(PP),32:GOTO 410
490 PO=PO-A(PP):GOTO 410
```

Concurso

Viene de la página anterior

```
500 NP=NP+2:PICAS=NP:POX=PO:TP$=TI$:GOTO 305
999 :
1000 REM    ******** FIN DE PARTIDA ****
1001 :
1005 PRINT"S3QQQQQQQQ]]]]]]]]]]]SE TERMINO EL TIEMPO■"
1010 PRINT"S3QQQQQQQQQQQQQQQQ]]]]]]]]]]QUIERE OTRA PARTIDA■";
1020 INPUT"]]SI]]]]]";RE$
1030 IF LEFT$(RE$,1)="S" THEN RUN
1040 PRINT"J":END
8999 :
9000 REM      ******** SONIDO ************
9001 :
9005 POKE 54277,9:POKE 54278,0
9010 POKE 54273,120:POKE 54272,32:POKE 54276,33
9020 FOR T=1 TO 25:NEXT:POKE 54276,32
9030 RETURN
9999 :
10000 REM      ******* INSTRUCCIONES *****
10001 :
10002 PRINT"J"
10010 PRINT"Q]]]]]]]]]]]]]]R7LABERINTO DE PICAS"
10020 PRINT"QQ]]]]]]QUIERE  INSTRUCCIONES";
10030 INPUT"]]N]]]]";R$
10040 IF LEFT$(R$,1)="S" THEN GOTO 10060
10050 RETURN
10060 PRINT"SQQQQQ
10070 PRINT"J  SU JUGADOR ES EL *. EL JUEGO CONSISTE"
10080 PRINT"QEN ELIMINAR TODAS LAS PICAS DEL LABE-"
10090 PRINT"QRINTO EN MENOS DE DOS MINUTOS."
10100 PRINT"Q  EN LA PRIMERA PANTALLA HAY 10 PICAS."
10110 PRINT"Q  EN CADA UNA DE LAS SIGUIENTES HAY"
10120 PRINT"QDOS PICAS MAS QUE EN LA ANTERIOR."
10130 PRINT"Q  POR CADA PANTALLA TIENE DOS MINUTOS"
10140 PRINT"QEXTRA, (UNO SI SE ELIGE DIFICULTAD D)"
10150 PRINT"QQUE SE LE SUMA AL TIEMPO QUE LE QUEDABA"
10160 PRINT"]]]]]]]]]]]]]IQAPULSE UNA TECLA PARA SEGUIR7"
10170 GET A$:IF A$="" THEN GOTO 10170
10180 PRINT"JQQ]]]]]]]]]]]]]]R7LABERINTO DE PICAS"
10190 PRINT"QQ  PARA MOVER SU JUGADOR UTILICE LAS SI-"
10200 PRINT"QGUIENTES TECLAS:"
10210 PRINT"Q]]]]]]]]]]]]]]]]Q - ARRIBA"
10220 PRINT"Q]]]]]]]]]]]]]]]]A - ABAJO"
10230 PRINT"Q]]]]]]]]]]]]]]]]: - DERECHA"
10240 PRINT"Q]]]]]]]]]]]]]]]]L - IZQUIERDA"
10243 PRINT"Q  PULSAR SOLO PARA CAMBIAR DE DIRECCION"
10245 PRINT"Q  PARA EMPEZAR ESPERE A QUE SE CREE"
10247 PRINT"QEL LABERINTO."
10250 PRINT"Q]]]]]IAPULSE CUALQUIER TECLA PARA EMPEZAR"
10260 GET A$:IF A$="" THEN 10260
10270 RETURN
19999 :
20000 REM ** ELECCION DE DIFICULTAD **
20001 :
20010 INPUT"JQQQQQQQQQQQ]DIFICULTAD (F=FACIL/D=DIFICIL) ";V$
20020 IF LEFT$(V$,1)="F" THEN DI=200:RETURN
20030 IF LEFT$(V$,1)="D" THEN DI=100:RETURN
20040 GOTO 20000
```

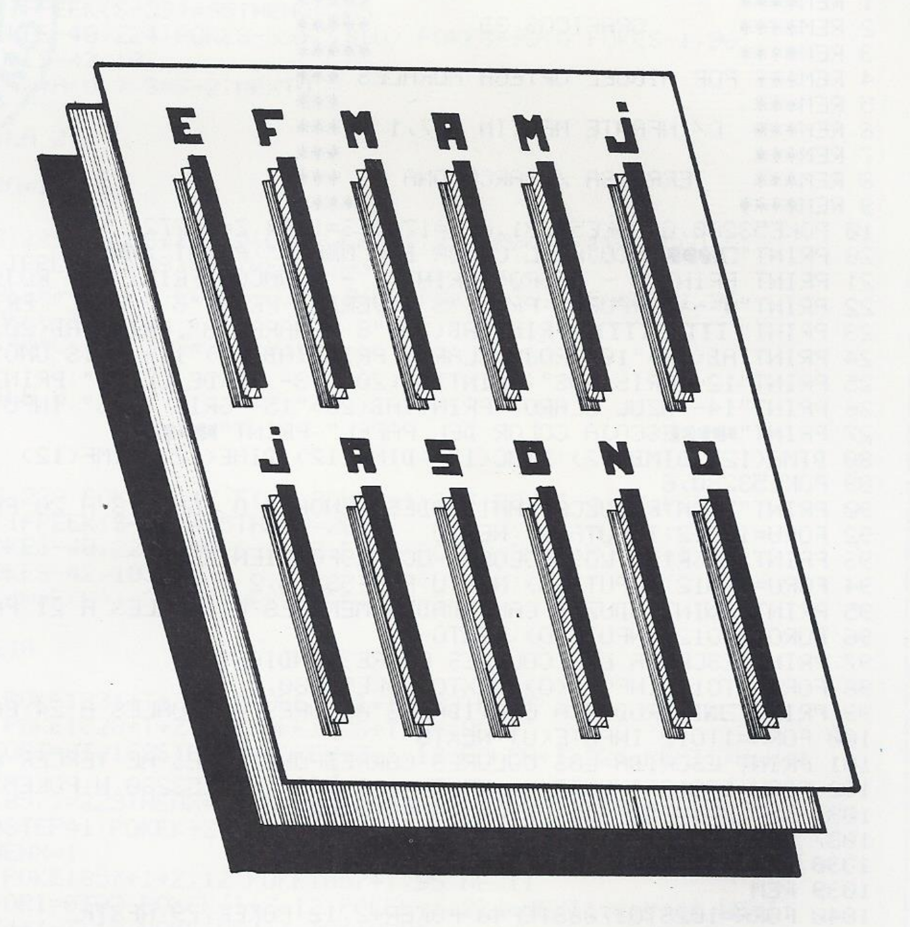
PROGRAMA COMPROBADO commodore Magazine

PREMIADO CON 5.000 PESETAS

Concurso

Gráficos 3 D

CBM 64



Desde Barcelona, Miguel Ortega Morales, envía a concurso este interesante programa para dibujar gráficos de barras en tres dimensiones.

El programa pregunta en primer lugar por el color del marco y del papel. A continuación hay que introducir los valores correspondientes a cada una de las barras. Hay doce barras por hilera, ya que se supone que en cada hilera se representan los resultados de un año, de modo que cada una de las doce barras se corresponde con uno de los meses del año. Los valores que se asocian a cada uno de los meses sólo pueden ir de cero a 20, ya que así está previsto en el programa.

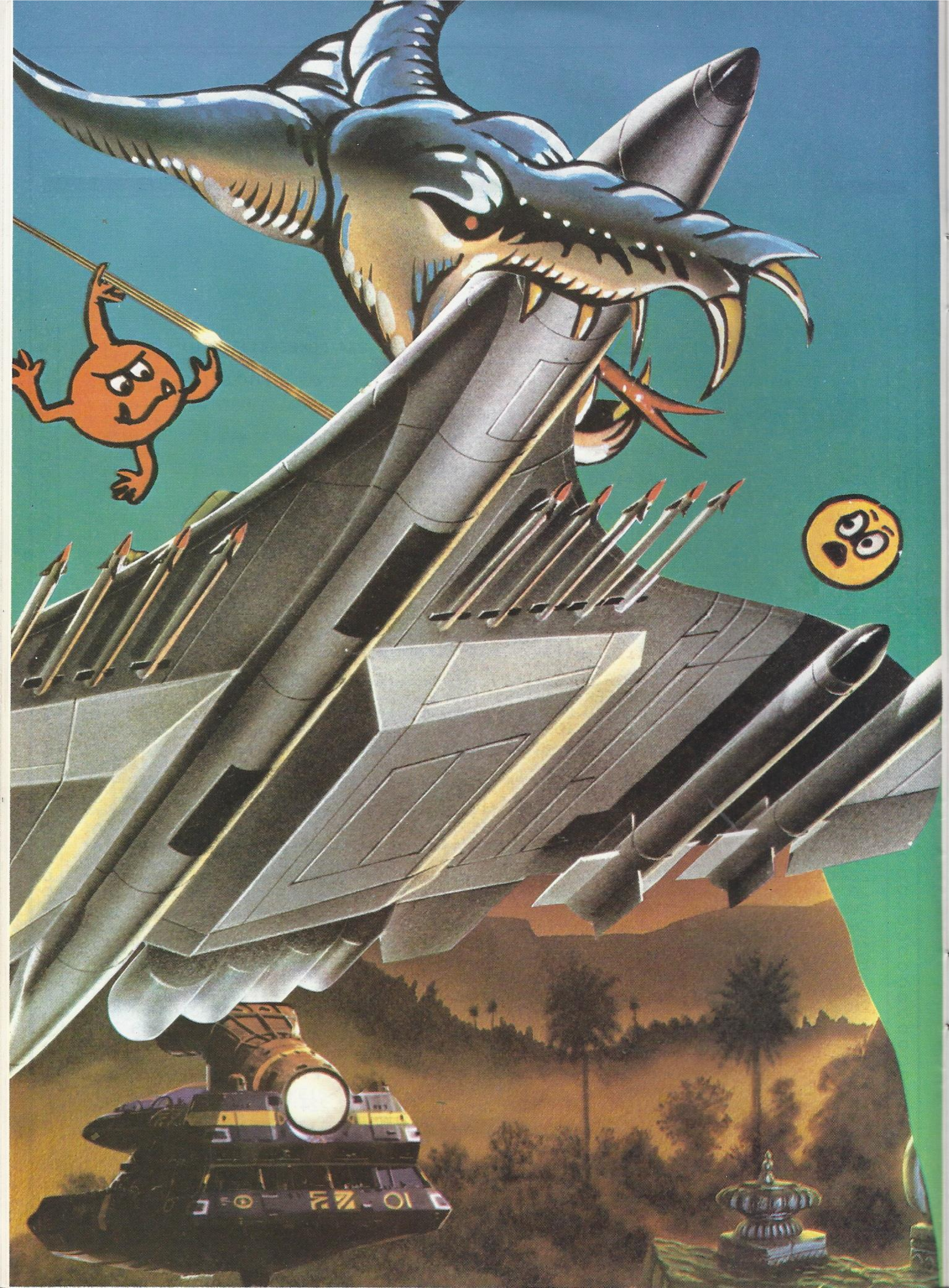
Después de introducir los valores correspondientes a los doce meses de la primera fila, se procede a introducir los valores correspondientes a los colores de cada una de las barras, teniendo en cuenta que el primer color introducido se corresponde con la barra situada más a la derecha.

A continuación se introducen los doce valores que corresponden a la representación del segundo año, con sus colores respectivos y por último hay que hacer lo mismo con el último año, que es el que viene representado por las barras más próximas a nosotros. Miguel, el autor del programa, aconseja que tanto para el color del fondo como para el color del papel se escoja el negro, con el fin de obtener los mejores resultados posibles y poder asombrar a nuestros amigos con unos maravillosos gráficos tridimensionales y a todo color con nuestro Commodore 64.

```
1 REM****                      *****
2 REM****     GRAFICOS 3D      *****
3 REM****                      *****
4 REM*** POR MIGUEL ORTEGA MORALES ***
5 REM***                        ***
6 REM***  C/INFANTE MARTIN 1,7,1  ***
7 REM***                        ***
8 REM***   TERRASSA / BARCELONA   ***
9 REM****                      ****
10 POKE53280,0:POKE53281,0:C=1749:S=1854:Z=54272
20 PRINT"█████ESCOJA EL COLOR DEL MARCO":PRINT"████                                   █"
21 PRINT:PRINT"0 - NEGRO":PRINT"1 - BLANCO":PRINT"2 - ROJO":PRINT"3 - CYAN"
22 PRINT"4 - PURPURA":PRINT"5 - VERDE":PRINT"6 - AZUL":PRINT"7 - AMARILLO"
23 PRINT"TTTTTTTTT":PRINTTAB(20)"8 - NARANJA":PRINTTAB(20)"9 - MARRON"
24 PRINTTAB(20)"10- ROJO CLARO":PRINTTAB(20)"11- GRIS UNO":PRINTTAB(20)
25 PRINT"12- GRIS DOS":PRINTTAB(20)"13- VERDE CLARO":PRINTTAB(20)
26 PRINT"14- AZUL CLARO":PRINTTAB(20)"15- GRIS TRES":INPUTN
27 PRINT"████ESCOJA COLOR DEL PAPEL":PRINT"█████                                 █":INPUTY
80 DIMA(12):DIMB(12):DIMC(12):DIMD(12):DIME(12):DIMF(12)
89 POKE53280,6
90 PRINT"██INTRODUZCA CANTIDADES MENORES O IGUALES A 20 PARA LA PRIMERA FILA█"
92 FORU=1TO12:INPUTA(U):NEXTU
93 PRINT"ESCRIBA LOS COLORES CORRESPONDIENTES"
94 FORU=1TO12:INPUTB(U):NEXTU:POKE53280,2
95 PRINT"██INTRODUZCA CANTIDADES MENORES O IGUALES A 21 PARA LA SEGUNDA FILA█"
96 FORO=1TO12:INPUTC(O):NEXTO
97 PRINT"ESCRIBA LOS COLORES CORRESPONDIENTES"
98 FORO=1TO12:INPUTD(O):NEXTO:POKE53280,5
99 PRINT"██INTRODUZCA CANTIDADES MENORES O IGUALES A 24 PARA LA TERCERA FILA█"
100 FORU=1TO12:INPUTE(U):NEXTU
101 PRINT"ESCRIBA LOS COLORES CORRESPONDIENTES AL TERCER ANO"
102 FORU=1TO12:INPUTF(U):NEXTU:PRINT"█":POKE53280,N:POKE53281,Y:FORI=0TO2500
103 NEXT:PRINT"█":GOSUB1040:GOSUB2000:END
1037 REM
1038 REM * DIVISIONES *
1039 REM
1040 FORR=1028TO1788STEP40:POKER+Z,12:POKER,79:NEXTR
1041 FORR=1060TO1820STEP40:POKER+Z,12:POKER,80:NEXTR
1042 FORT=1189TO1219:POKET+Z,12:POKET,99:NEXTT
1043 FORT=1389TO1419:POKET+Z,12:POKET,99:NEXTT
1044 FORT=1589TO1619:POKET+Z,12:POKET,99:NEXTT
1046 POKE1185+Z,12:POKE1185,36:POKE1186+Z,12:POKE1186,51:POKE1187+Z,12
1047 POKE1187,48
1048 POKE1385+Z,12:POKE1385,36:POKE1386+Z,12:POKE1386,50:POKE1387+Z,12
1049 POKE1387,48
1050 POKE1585+Z,12:POKE1585,36:POKE1586+Z,12:POKE1586,49:POKE1587+Z,12
1051 POKE1587,48
1052 POKE1785+Z,12:POKE1785,36:POKE1787+Z,12:POKE1787,48
1096 REM
1097 REM    COLUMNAS FILA1
1098 REM
1099 FORU=1TO12
```




```
1100 FORI=0TOA(U)-1
1200 POKES+Z,B(U):POKES,229:POKES+1+Z,B(U):POKES+1,224:POKES-41+Z,B(U)
1210 POKES-41,95:Q=223:IFPEEK(S-39)=95THENQ=205
1300 POKES-40+Z,B(U):POKES-40,224:POKES-39+Z,B(U):POKES-39,Q:POKES-1,96
1310 POKES-42+Z,B(U):POKES-42,103
1400 S=S-40:NEXTI:S=S+(40*A(U)):S=S-2:NEXTU
1497 REM
1498 REM     COLUMNAS FILA 2
1499 REM
1500 S=1895:Z=54272:FORO=1TO12
1510 FORI=0TOC(O)-1
1520 POKES+Z,D(O):POKES,229:POKES+1+Z,D(O):POKES+1,224:POKES-41+Z,D(O)
1530 POKES-41,95:Q=223:IFPEEK(S-39)=95THENQ=205
1540 POKES-40+Z,D(O):POKES-40,224:POKES-39+Z,D(O):POKES-39,Q:POKES-1,96
1550 POKES-42+Z,D(O):POKES-42,103
1560 S=S-40:NEXTI:S=S+(40*C(O)):S=S-2:NEXTO
1597 REM
1598 REM     COLUMNAS FILA 3
1599 REM
1600 S=1936:FORU=1TO12
1610 FORI=0TOE(U)-1
1620 POKES+Z,F(U):POKES,229:POKES+1+Z,F(U):POKES+1,224:POKES-41+Z,F(U)
1630 POKES-41,95:Q=223:IFPEEK(S-39)=95THENQ=205
1640 POKES-40+Z,F(U):POKES-40,224:POKES-39+Z,F(U):POKES-39,Q:POKES-1,96
1650 POKES-42+Z,F(U):POKES-42,103
1660 S=S-40:NEXTI:S=S+(40*E(U)):S=S-2:NEXTU:RETURN
1997 REM
1998 REM            BANDEJA
1999 REM
2000 FORT=0TO162STEP41:POKE1831+T+Z,12:POKE1831+T,223:NEXT
2001 FORT=0TO162STEP41:POKE1828+T+Z,12:POKE1828+T,95:NEXT
2002 H=1829:FORT=1TO2:FORI=0TO162STEP41:POKEH+Z+I,12:POKEH+I,224:NEXTI:H=H+1
2003 NEXTT
2004 Y=224:X=0:IFPEEK(1857)=223THENX=1:IFPEEK(1857)=223THENY=223
2005 K=1861:FORI=0TO120STEP41:POKEK+Z+I,12:POKEK+I,223:NEXT:POKE1979+Z,12
2006 IFPEEK(1857)=224THENX=1
2007 FORI=0TO120STEP41:POKE1857+I+Z,12:POKE1857+I,95:NEXTI
2009 K=1858:FORT=1TO3:FORI=0TO2:POKEK+I+Z,12:POKEK+I,224:NEXTI:K=K+41:NEXTT
2013 FORI=0TO29:POKE1954+I+Z,12:POKE1954+I,224:NEXT
2014 IFX=1THENGOSUB3000
2015 IFX=0THEN2115
2016 GOTO2016
2115 RETURN
2116 GOTO2116
2997 REM
2998 REM ADECUAR LA BANDEJA A CANTIDADES MINIMAS EN LAS FILAS DE DICIEMBRE
2999 REM
3000 POKE1898+Z,12:POKE1898,224:POKE1938+Z,12:POKE1938,224:POKE1939+Z,12
3001 POKE1939,224:POKE1857+Z,F(1):POKE1857,Y:RETURN
```



¡¡Exclusivo para lectores de 



CLUB DE SOFTWARE



Ha seleccionado para usted cuatro excelentes programas de juego para su ordenador **VIC-20. COMMODORE 64.**

Diviértase con ellos y, **además, ahorre dinero** con esta oferta única y exclusiva para los lectores de esta revista.



HIDEOUS BILL

¿Está Bill (es decir, tú) dispuesto a luchar contra los gigantes para salvar a Greta, su (tú) único y verdadero amor? Si no lo logras, será comida por las hormigas. Y no olvides el peligro del barro venenoso, a no ser que lleves contigo una tableta de antídoto. Puedes valerte de los arpones, pero durarán poco. Rompiendo paredes abrirás caminos para la fuga. Al final, el amor y una sorpresa.
Precio: 1.550 ptas.

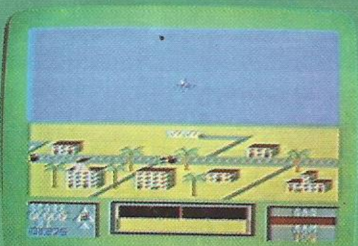


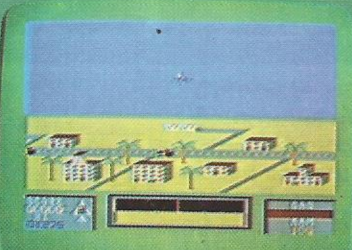
AMBUSH

Estás pilotando una nave espacial y tienen que cargar combustible de una nave nodriza. En el camino serás atacado. Puedes utilizar tu cañón de rayo laser, pero cuando un atacante es alcanzado no muere: se divide en dos y tu problema se multiplica. No acaban allí tus dificultades: tienes que esquivar (o destruir) sus bombas de protones. Hay siete, ¡siete! oleadas de enemigos. Tal vez te convenga atracar en la nave nodriza para coger un cañón laser más potente.
Precio: 1.550 ptas.

FALCON PATROL

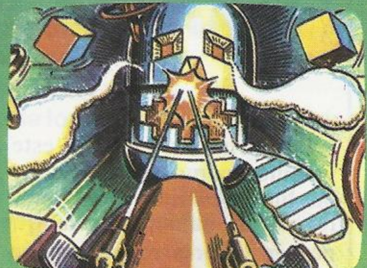
Como único superviviente de la patrulla Halcón, te toca a tí conseguir que los reactores atacantes no destruyan las instalaciones petroleras y tus bases de aprovisionamiento. Estás armado con cien misiles y un radar muy sofisticado en pantalla. Si te quedas sin combustible, podrás aterrizar verticalmente, pero mientras estés en tierra serás vulnerable a la metralla. Cuanto más tiempo sobrevivas, más enemigos vendrán a por tí.
Precio: 1.550 ptas.





BITMANIA

Sorpresa. Estás dentro del bus, da datos de tu 64 y debes batallar contra un aluvión de neuronas. Tu tarea, alcanzar el núcleo del sistema. Pero para defenderse a sí misma la CPU envía contra tí sus ejércitos de bits. Tú puedes destruirlos, pero cuando lo consigues se transforman en pods, que te impiden avanzar al siguiente nivel. Y además, cada 8,5 segundos la CPU enviará otro bit para reemplazar a cada uno destruído. Si tienes templados los nervios, no caerás en el pánico.
Precio: 1.550 ptas.



Oferta especial 4x3

Usted puede comprar cualquiera de estos cassettes a su precio individual de 1.550 Ptas. Pero también puede adquirir el lote de 4 por sólo 4.650 Ptas. Y AHORRARSE 1.550 Ptas. O sea, cuatro cassettes por el precio de tres. Aproveche esta oferta limitada.

Recorte y envíe este cupón HOY MISMO a:
INFODIS, S.A.
Bravo Murillo, 377-5.º-A
28020 MADRID

CUPON DE PEDIDO

Envíenme a mi domicilio, al precio de 1.550 Ptas. cada una, más 95 Ptas. de gastos, la cassette o cassettes que le indico a continuación, marcando con una (X) en el casillero correspondiente. Queda bien entendido que pidiendo las 4 cassettes obtengo un ahorro de 1.550 Ptas., lo que me supone UNA CASSETTE GRATIS **(Sólo pagaré 4.650 Ptas. más 95 Ptas. por gastos de envío).**

☐ **HIDEOUS BILL** ☐ **AMBUSH** ☐ **FALCON PATROL** ☐ **BITMANIA**

☐ Las 4 cassettes, en las ventajosas condiciones de AHORRO para mí **(4.650 Ptas.).**

El importe lo abonaré:
POR CHEQUE ☐ **CONTRA REEMBOLSO** ☐ **CON TARJETA DE CREDITO** ☐
American Express ☐ **Visa** ☐ **Interbank** ☐ Fecha de caducidad ____

Número de mi Tarjeta ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

Nombre ____

Dirección ____

Ciudad ____ D.P. ____

Provincia ____

Cada envío llevará 95 Ptas. en concepto de gastos.



CAPITULO 2

EL MISTERIO DEL BASIC

En la primera parte de este artículo veíamos como el ordenador tomaba las líneas escritas en BASIC y transforma las **palabras clave,** que contienen en valores equivalentes y ocupan un solo byte, llamados *token.* Posteriormente, el **intérprete** los reconoce, ejecutándolos de manera más rápida que si la palabra hubiera estado almacenada con su longitud real.

Una vez más, al diseñar los sistemas informáticos, los proyectistas imitan a la naturaleza. Así, el ordenador cuando lee un programa nos imita a nosotros mientras leemos esta página. Comienza desde el principio, leyendo letra por letra (byte por byte), incluyendo a los espacios en blanco, interpretando el significado de cada agrupación de caracteres o carácter individual (léase también bytes, byte), hasta llegar al final.

Cuando damos la orden de RUN, o presionamos Return después de teclear una línea sin numerar, el ordenador lee una línea completa, comenzando por la de menor numeración, pasando a la siguiente una vez interpretada y ejecutada ésta. Puede ocurrir que no pase a la siguiente línea, sino a otra indicada por GOSUB o GOTO.

La línea resulta leída por orden, desde el comienzo al fin, exactamente igual que haríamos nosotros.

Un programa se almacena en la memoria del **64** ó el **Vic 20** de manera similar a como escribimos un texto en un folio de papel, utilizando para ello el área destinada a los programas BASIC, comprendida entre las direcciones 2048 ($0800) y 40959 ($9FFF).

Sigamos viendo qué ocurre cuando se teclea el comando RUN (seguido **por Return**), el sistema operativo lo **interpreta** como un comando directo **(no lleva número de línea)** y busca en su lista de palabras reservadas, para **encontrar** la dirección de la subrutina **que ejecuta** este comando. Finalmente es localizado en la dirección hexadecimal $A871. Como primeros pasos, el comando pone todos los punteros a comienzo del programa (luego veremos qué significa esto), restaurando también todos los punteros de subrutina y datos.

Como decíamos antes, se busca la primera línea de programa. Ahora hay que leerla y para ello se sigue un procedimiento, exactamente igual al que rige la lectura de una línea de texto impreso. De cara a esta fase, **Commodore** ha dispuesto en sus ordenadores una útil rutina bautizada como CHARGET, que es una abreviación de las dos palabras inglesas CHARacter GET, o lo que es igual, "consigue un carácter". Esta rutina, escrita en código máquina, forma parte del **intérprete,** pero no todo él está contenido en la ROM. En el caso de esta rutina ocurre una cosa peculiar. En un principio se halla contenida en la ROM, entre las direcciones 58274 ($E3A2)y 58297 ($E3B9), pero con ayuda de otra rutina es copiada en la página cero del **64** (direcciones

de memoria comprendidas entre cero y 255 ó \$00 y \$FF). La razón es simple: los datos contenidos en la ROM son inalterables. Sin embargo, mientras utilizamos CHARGET existen algunos bytes de la rutina que deben ser alterados, durante la interpretación de las sentencias BASIC; por ello se hace necesaria la utilización de memoria RAM. La copia de CHARGET desde la memoria ROM a las direcciones 115 (\$73) a 138 (\$8A) se realiza durante la inicialización del sistema, una vez que conectamos el ordenador.

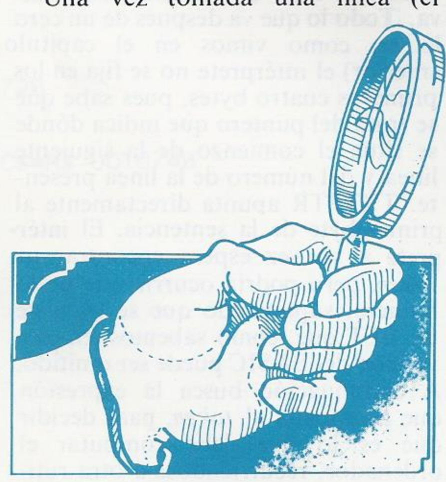
Estos bytes a los que acabamos de aludir son bautizados como punteros y su función no es otra que la de apuntar. Su misión es algo similar a lo que hacíamos cuando aprendimos a leer, señanlando con el dedo la siguiente letra de la línea que estábamos interpretando.

En la rutina CHARGET, el puntero es conocido como TXTPTR (TeXT PoinTeR) o puntero del texto BASIC, formado por dos bytes, pudiendo así "apuntar" hacia cualquiera de las direcciones que debe interrogar.

TXTPTR utiliza las direcciones \$7A y \$7B, conteniéndose en ellas cual es el siguiente carácter del programa en BASIC que debe ser leído. Como de costumbre, en la primera dirección se almacena el byte de menor peso y en el siguiente el de mayor.

Antes de pasar a describir en detalle la rutina CHARGET, veamos algo más sobre el mecanismo que sigue el **intérprete** para que el programa vaya siendo ejecutado.

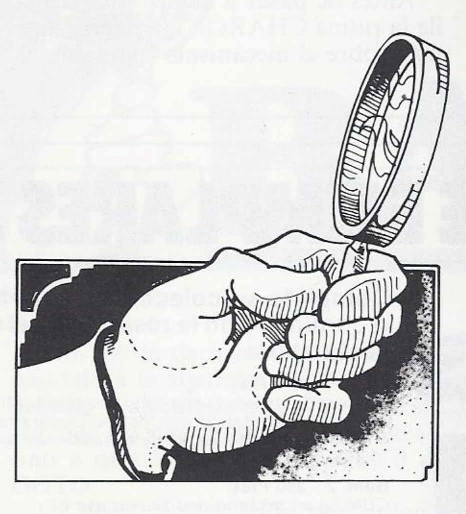
Una vez tomada una línea (el



intérprete sabe que es una línea nueva. Todo lo que va después de un cero lo es, como vimos en el capítulo anterior) el intérprete no se fija en los primeros cuatro bytes, pues sabe que se trata del puntero que indica dónde se sitúa el comienzo de la siguiente línea y del número de la línea presente. TXTPTR apunta directamente al primer byte de la sentencia. El **intérprete** a su vez espera encontrar un *token,* pero podría ocurrir que no lo hubiera, suponiendo que se trata de un LET que, como sabemos, en esta versión del BASIC puede ser omitido. A continuación busca la expresión que acompaña al *token,* para decidir qué es lo que debe computar el ordenador, recurriéndose a otra rutina que se llama *Evaluate Expresion* y está situada en la dirección hexadécimal $AD9D. Su trabajo consiste en evaluar la expresión, sea numérica o una cadena, que hay escrita desde la dirección que alberga TXTPTR hasta el primer carácter delimitador que encuentre, sea un punto y coma, comillas, un cero, dos puntos, paréntesis, etc.

Ocurre que en ninguna de las versiones del BASIC de **Commodore** está prevista la comprobación de sintaxis durante la introducción del programa, por lo que pueden escribirse sentencias que carezcan de sentido. Es durante la ejecución del programa cuando se lleva a cabo este proceso. Puede existir un error de sintaxis en la escritura de una **palabra clave** o en la expresión que le acompaña. Esto va realizándose a medida que CHARGET va desmenuzando la línea. Igualmente, una vez ejecutada la sentencia y devuelto el control al **intérprete,** éste comprueba si a continuación hay un cero, que indica el final de la línea, o dos puntos, que indican el final de una sentencia, produciendo un error de sintaxis en caso contrario.

Recordemos el proceso: el **intérprete** sabe que se encuentra ante un *tokem* siempre que el byte sea mayor de 128, a no ser que se encuentre formando parte de una cadena, que irá entrecomillada. Posteriormente, como decíamos antes, localiza la rutina de ejecución correspondiente a ese *token,* pasándole el control a la misma, que decidirá sobre lo que tiene que hacer y dónde debe buscar la información necesaria. Posteriormente devuelva el control al intérprete que localiza la siguiente sentencia, situada en la misma o en otra línea. En el segundo caso, se localiza la nueva línea, mediante la subrutina de búsqueda de línea.



Entremos ya en la descripción del funcionamiento detallado de la rutina CHARGET, que dispone de dos puntos de entrada, uno es la propia CHARGET, que incrementa a TXTPTR para indicarle cuál es el próximo carácter a leer, y el otro es CHARGOT (Carácter obtenido), que lee el carácter y no afecta al cotenido de TXTPTR. Decíamos antes que cuando comenzamos a ejecutar un programa, el **intérprete** se encarga de que TXTPTR apunte hacia el comienzo del programa.

En este momento es conveniente fijar la atención en la rutina escrita en código máquina en la figura 1. No es ni más ni menos que nuestra vieja conocida CHARGET. Una importante misión que lleva a cabo es la de ignorar los espacios en blanco que introducimos cuando escribimos una línea de programa. Es fácil comprobar que éstos no afectan para nada a la correcta ejecución de la línea, tanto si hay de más como si los omitimos totalmente.

Entre las direcciones $73 a $77 se lleva a cabo la operación de actualizar el puntero existente en las direcciones $7A y $7B. Su mecanismo de funcionamiento es el siguiente: E6 7A incrementa el byte de menor peso del "puntero de caracteres". A continuación D0 02 comprueba si este byte de menor peso es igual a 255 ($FF) y si la respuesta es afirmativa, pasa a la siguiente orden, E6 7B, que incrementa el byte de mayor peso situado en $7B. En caso contrario salta dos bytes hacia adelante, pasando directamente a CHARGOT, donde toma el carácter situado en la dirección contenida en el puntero referida al texto fuente (TXTPTR), que siempre va siendo incrementado en los pasos anteriores del programa.

CHARGOT se ocupa de cargar en el acumulador del microprocesador el byte contenido en la dirección de memoria indicada por los bytes $7A y $7B (TXTPTR), a partir del cual es comparado con 3A, byte correspondiente al carácter dos puntos (:), saltando al final de la rutina en caso afirmativo, pues ha terminado la presente sentencia. De no ser así, se procede a comparar el carácter del texto BASIC con el byte $20, cuyo fin es determinar si estamos ante un espacio en blanco, si es afirmativo, volvemos al principio de la rutina CHARGET, obviándose de esta manera el espacio. Después se activan los flags para según que tipo de carácter, acabándose de ejecutar la subrutina, con 60 (RTS = vuelve de la subrutina).

La operación de esta rutina se repite una y otra vez, hasta que todo el programa ha sido barrido, desde el principio al fin.

Aclaración

En el listado del MON64, publicado en el número anterior, existe un error que impedía obtener un volcado de la memoria en impresora. Para solucionarlo seguir estos pasos:

1) Cambiar la línea 745 por:
745 IF PEEK (+197) = 1 THEN ME=AC

2) Cambiar la numeración de la línea 7340 pasándola a 7347.

3) Cambiar la numeración de las líneas 700 y 701 pasándolas a 7340 y 7345.

4) Borrar las líneas 700 y 701.

```
640 GOSUB7300
680 IFA$="M"THEN 770
705 FORME=CM TO AC
710 IFHX=1THEND=ME:GOSUB5500:P$=H$:GOTO720
715 P$=STR$(ME)
720 PRINTP$;"   ";
730 C=PEEK(ME):IFHX=1THEND=C:GOSUB5500:C$=H$:GOTO740
735 C$=STR$(C)
740 PRINTC$
745 IFPEEK(197)=1THEN ME=AC
746 IFPEEK(197)=60THENGOTO746
750 NEXT
READY.

7330 GOTO7320
7340 PRINT"■PULSA 'RETURN' PARA ACABAR""
7345 PRINT"O LA BARRA PARA PARAR.■■■""
7347 OPEN1,CH:CMD1
7350 RETURN
```

P.: El club Ensenada de Santander ha puesto en marcha una asociación de usuarios **Commodore.** Disponen de ordenadores **CBM 64** y **Vic 20**, dotados con diversos periféricos. Su actividad se centra en el intercambio de programas y experiencias, perfeccionamiento de lenguajes y uso de logo. Está destinado más que nada a los alumnos de 2.ª etapa de BUP y EGB de Santander. Las señas son: calle Santa Lucía, n.° 1, 1.ª y el teléfono es el 22 59 92.

Club Ensenada (Santander)

R.: *Pues ánimo, los "commodoreros" de la zona están de enhorabuena, en este club os podréis reunir y cambiar impresiones. Cuando estéis a tope, escribirnos y contarnos vuestros avances y programas nuevos.*

P.: Soy poseedor de un **Vic-20** y reciente lector de su también nueva revista, la cual me gusta y me interesa mucho. Les deseo lo mejor para que el camino de su revista no se entorpezca ni se detenga nunca, ya que una revista amena y libre de ataduras es lo que nos hace falta a nosotros, los poseedores de ordenadores de **Commodore**, con ganas de exprimirlos y estrujarlos al máximo, para sacar u obtener unos resultados meritorios y satisfactorios...

También les contaré que por mi cabeza rondan varios programas para realizar, de los cuales si alguno de ellos lo creo oportuno, se los mandaré, para participar en el concurso de programas, de gran acierto califico yo este tipo de iniciativas, pues promocionan enormemente el trabajo con el ordenador, con sus consecuentes beneficios para los nuevos programadores.

Jordi Grau. Barcelona

R.: Muchas gracias por las reflexiones y elogios vertidos en tu misiva. Manos a la obra con esos programas, que lo que te ronda por la cabeza no se puede pasar desde ella a la impresora, y a ver si hay suerte.

Por nuestra parte el ánimo que nos trasmites nos ayudará a esforzarnos en tener una revista cada día más amena.

P.: Tengo un **C-64.** Supongo que aquellos que lo disfrutéis en Santa Cruz de Tenerife sabréis las tremendas dificultades para hacerse con libros, programas, etc. Os propongo a los que queráis, poneros en contacto conmigo y organizar un pequeño grupo para intercambiar todo tipo de ideas y experiencias. Creo que nos vendrá bien.

Me llamo Fernando Martínez. Escribir a Gral. Fanjul, 30, 3.° izquierda o llamarme de lunes a viernes de 8 a 3 al teléfono 28-50-01.

R.: Pues ya lo sabéis los *commodoremen* de la preciosa isla.

P.: Mi pregunta es la siguiente; es necesario realizar modificaciones para que un programa escrito para un **Vic-20** se puede introducir a un Commodore 64, o la impresión directa es suficiente. En el caso que se tuviesen que realizar modificaciones, les rogaría me las indicasen; en el caso de que esto les fuera imposible, les rogaría me remitieran bibliografía sobre el tema.

Sugiero que sería interesante que dedicaran algunos artículos a realizar estudios comparados de diferencias entre programaciones de diversos ordenadores, tales como el **Commodore 64**, el **Vic-20** y el **Spectrum.**

Leopoldo Bello
Barcelona

R.: En el número 2 de **Commodore Magazine** dedicamos algún espacio a comentar las diferencias con que nos encontramos a la hora de utilizar en un ordenador los artículos desarrollados en el otro. Sin embargo le aclaramos brevemente. Cuando el programa está escrito íntegramente en un BASIC que no hace referencia a determinadas direcciones de memoria (con POKE o PEEK), las probabilidades de que funcione sin modificaciones son muy altas. Ahora bien, cuando se entra en programas que utilizan desarrollos gráficos *(sprites* con el C-64) o música, es improbable que el programa sea bivalente.

De cualquier manera estamos trabajando en otro artículo más explícito, que será publicado cuando se complete, sirviendo de valiosa ayuda a esta traducción de programas.

En cuanto a su última sugerencia, parece que nos haya leído el pensamiento. Efectivamente, una de nuestras previsiones futuras se encamina hacia establecer comparaciones enre la diversificada oferta del mercado.

P.: SOy propietario de un **Commodore PET 2001** desde hace unos cuatro años. Esto lo adquirí de segunda mano, y la comprarlo sólo me entregaron una fotocopia del manual de instrucciones y una cinta con cuatro programas de juegos.

Sinceramente, me gustaría poderle sacar más partido al ordenador, pero mis conocimientos en la rama de la programación son insuficientes para desarrollar un program de utilidad (matemática, física, doméstica, ...).

Al suscribirme a **"Commodore Magazine"** esperaba poder adquirir de esta revista una serie de programas y documentos interesantes aplicables a "mi **PET**", pero no ha sido así. La información y programas que he podido encontrar son simplemente aplicables al **Vic-20** y **CBM 64.**

En definitiva, solicito de su amabilidad información sobre el modo de encontrar programas e información para mi ordenador.

Pedro J. Boj
Alicante

R.: Esta serie de ordenadores de **Commodore** fue de una gran aceptación en Europa. Sin embargo el parque existente en España tiene unas dimensiones no comparables a los correspondientes a los otros dos mode de los que nos menciona. Por otro lado es usted el primer usuario de este modelo que nos escribe, por lo que hemos pensado que adaptarían algunos programas de los destinados a **Vic-20** y **CBM 64.** En muchos casos, con ayuda del manual, no es tan difícil. Sin embargo, en un futuro artículo trataremos sobre este tema.

Este veterano modelo fue de gran implantación, igual que el **Apple II,** en determinados círculos profesionales y universitarios, por lo que es de prever que existan interesantes aplicaciones desarrolladas por ellos.

Logo para el C-64

BREVE HISTORIA

El **LOGO** fue desarrollado en 1968 por **Seymour Papert** en el **Laboratorio de Inteligencia Artificial del M.I.T.**, el célebre Instituto Tecnológico de Massachusetts. Durante toda la década de los 70 este lenguaje, derivado parcialmente del **LISP** (LISt Processing, lenguaje con tratamiento de listas), era soportado únicamente en grandes ordenadores, debido a la complejidad y sofisticación.

A partir de 1975 se produce el adveminiento y "boom" de los microordenadores, y en 1979 el **Grupo Logo** del **M.I.T.** inicia un esfuerzo de adaptación para lograr implementar este lenguaje sobre microcomputadores. En 1981 y bajó la respectiva supervisión del propio **S. Papert** y de **Harold Abelson** se prepararon las dos primeras versiones, para el **Texas Instruments TI 99/4** y el **Apple II.**

El **TI Logo** permitía el uso de "duendes" *(sprites)* a diferencia del **Apple Logo,** que a cambio, trabajaba con una aritmética de coma flotante (no solamente con números enteros), disponía de unas instrucciones primitivas de tratamiento de listas más potentes y gestionaba el "espacio de trabajo" con un empaquetamiento más denso que ahorraba memoria.

Durante los años 1982 y 1983 todas las grandes compañías (**IBM, TANDY, ATARI, COMMODORE,...**) han incorporado este lenguaje a sus sistemas informáticos.

LOGO EN EL CBM 64

Commodore con licencia de **M.I.T.** y **TERRAPIN** presenta una excelente versión para el ordenador personal **CBM 64,** realizada bajo la dirección de **Leigh Klotz Jr.**, que había participado en los dos primeros proyectos reseñados anteriormente. El lenguaje, de precio moderado (1.600 pts.), está disponible en España desde principios de 1984 (el **Logo** de **Apple** se comercializó un año antes y el microordenador **TI 99/4** es prácticamente desconocido en nuestro país). Se compone de dos discos flexibles: uno (LOGO) es el lenguaje y el otro (LOGO UTILITIES) es un amplio paquete de programas de demostración y con procedimientos suplementarios de gran utilidad, junto con un completo manual de fácil lectura (aunque todavía esté en inglés). También sería aconsejable un formato de cintas para el Datasette, a fin de extender el potencial número de usuarios, si bien las tareas de carga y búsqueda de programas serían algo lentas y engorrosas.

Esta versión de **Commodore** ha venido precedida en el tiempo por adaptaciones, como **TURTLE GRAPHICS II** (en cartucho ROM y con un coste similar de 15.000 pts.), que son sublenguajes incompletos, sucedáneos del **Logo,** de prestaciones muy inferiores e implementación mucho menos acertada y cuya exclusiva ventaja reside en no exigir unidad de disco para su utilización.

IMPLEMENTACION

Aprovechando la cualidad del **CBM 64,** que permite remodelar todo el mapa de memoria con sus 64 KB, se ha obtenido un **Logo** de alta calidad, con "duendes", aritmética decimal con números racionales, tratamiento de listas y palabras muy elaborado, almacenamiento compacto, instrucciones musicales, variables locales, ...

La redistribución asigna 28 KB al intérprete **Logo,** 1 y 8 KB para las pantallas de texto y gráfica (disponibles en todo tiempo y mezclables en proporciones variables), 4.5 KB para cadenas de texto, 4 KB para la pila de recursión (la recursividad es una de las propiedades más notables del **Logo** y se comentará al final) y 7.5 KB para otros apartados. Los restantes 12 KB para la "memoria de nodos" (nodespace) son suficientes, dada la concentración del sistema empleado ("listas de listas"), las rápidas y periódicas reorganizaciones para retirar lo no-permanente (*"garbage* collection" = recogida de basura) y un uso inteligente, que corresponde al programador del periférico monofloppy **1541.**

¿A QUIENES PUEDE INTERESAR EL LOGO?

El periodo de tiempo durante el cual el **Logo** ha estado disponible para un microordenador es reducido, pero permite prever la existencia de un amplio sector de futuros usuarios. Estos podrían clasificarse en cuatro grupos:

a) Los programadores EXPERTOS, que conocen y dominan el BASIC, pero quieren pasar a un "lenguaje estructurado", rápido de ejecución, con listados legibles, ... También quienes explotar el tratamiento de listas, la recursividad, la facilidad de creación y uso de "procedimientos", la comodidad para trabajar en código máquina, la ventaja de ser un lenguaje abierto, extensible, que cada uno construye a la medida de sus necesidades e intereses (pudiendo generar por módulos un lenguajes personal, único), ...

b) Los programadores NOVATOS, que pueden encontrar el BASIC (plagado de covencionalismos artificiosos (por ejemplo: "se debe especificar si las variables son alfanuméricas, enteras, ..."), y se desaniman ante un programa que no funciona por olvido de alguna nimiedad. **Logo** es

un lenguaje más inteligente, más evolucionado, de "más alto nivel que el BASIC, es decir, más próximo al lenguaje natural, más fácil de aprender, permitiendo una comunicación con el ordenador más flexible, más libre de "claves". Además el **Logo** es muy recomendable como primer lenguaje, porque facilita el paso posterior a los demás lenguajes. Lo ideal es pasar de **Logo** a BASIC, Pascal, ..., aunque en el proceso recíproco el conocimiento de estos lenguajes también simplifica notablemente el aprendizaje del **Logo**.

c) Los NIÑOS (¡atención los padres!) que encuentran en el **Logo** la posibilidad de comprender, dirigir y dominar la tecnología más avanzada que existe hoy en día. El **Logo** puede lograr una auténtica "alfabetización informática" *(computer literacy)*. El **Logo** desmitifica la Informática y desvanece infundados temores y esperanzas. Es el único lenguaje de programación recomendable para aprender en edades escolares, porque no condiciona ni estructura el pensamiento de modo artificial (el BASIC puede ser nefasto, habitúa a hacer piruetas mentales con sus GOTOs). El **Logo** es el único lenguaje nacido de la mano de un especialista en "Psicología evolutiva" **(S. Papert)**. Se debe resaltar que el **Logo** es un lenguaje idóneo para el aprendizaje y apto para los niños, pero no es sólo de niños. Como dice **Horacio Reggini,** impulsor del **Grupo Logo** argentino, "el castellano es un lenguaje para niños, pero también lo es de científicos, poetas, ...".



Logo—

d) Los DOCENTES para quienes la opción **Logo** puede ser el modo óptimo de E.A.O. (enseñanza asistida por ordenador). Sin relación alguna con la "enseñanza programada", que responde a un esquema autoritario e implica un conocimiento reflejo, poco interiorizado, el **Logo** nace de una filosolfía explícita del piagetiano **Papert**, que see expone en sus delicioso libro "Desafío a la mente" (véanse las referencias bibliográficas 1 y 2). El **Logo** hace más hincapié en el aprendizaje del alumno que en la enseñanza del profesor Un aprendizaje activo: "aprender haciendo y enseñando a hacer al ordenador, que sabe menos que el escolar". La función del proesor cambia de transmisor de conocimientos a guía del aprendizaje. El protagonista es el colegial apoyado por su maestro, nunca el ordenador. El **Logo** permite crear microcosmos ricos y asimilables de geometría, lengua, música, dibujo, ciencias, ..., animándole a explorarlos y resdescubrirlos. Si para aprender inglés hay que ir a Inglaterra, para aprender matemáticas lo mejor es ir a Matematelandia, y este lugar puede resultar accesible con el **Logo**. El **Logo** favorece el conocimiento reflexivo y personalizado, lo cual no significa aislado, pudiendo trabajar en equipo (un lema de **Papert** es "prohibido no copiar"). Esta utilización de la Informática en la Educación no uniformizada a los colegiales. sino que les ayuda a afirmar su propia personalidad y mejorar su autoconocepto. En EE.UU., Reino Unido, Francia, .. la apuesta por el **Logo** parece definitiva. En España es el objetivo prioritario para las personalidades más ilustres de la Educación-Informática (E. García Camarero, A. Requena, ..) y los grupos más avanzados (GIES de Santiago, Cero de Valencia, Colegios Rula y S. Ignacio de Barcelona, ...).

PUESTA EN MARCHA DE LOGO

El lenguaje se carga fácilmente mediante las habituales sentencias BASIC:

LOAD "LOGO", 8

que carga un pequeño programa previo y RUN para llamar y cargar el programa principal. Transcurridos unos 90 segundos, aparece el mensaje:

COMMODORE LOGO
COPYRIGHT (C) 1982, 983 TERRAPIN, INC.
COPYRIGHT (C) 1981 MIT
WELCOME TO LOGO!
?

Desde este momento podemos

olvidarnos del arcaico, desorganizado algo perezoso BASIC. En este artículo se harán frecuentes referencias BASIC, evitándose entremezclar a otros lenguajes como **Lisp, Forth, Pascal, Ada** o **Modula,** con los cuales tiene **Logo** mayor analogía pero que son muy poco conocidos. En el próximo artículo sobre **Forth** si confrontaremos **BASIC-Logo-Forth,** y así sucesivamente hasta llegar a un estudio comparativo **BASIC-Logo-Forth-Pascal** ...) un conjunto de sentencias llamadas "primitivas". Agrupando éstas se obtienen "procedimientos", que son nuevas órdenes y pueden utilizarse a su vez (y eventualmente junto con otras primitivas) para definir nuevos procedimientos.

Para explicarlo mejor: imaginemos que queremos hacer en pantalla un tablero de ajedez (Figura 1) para jugar con un amigo o para ver partidas famosas representadas (Figura 2) en el TV paso a paso (¡no intente este programa en BASIC!). El programa sería símplemente el procedimiento AJEDREZ. Dado que AJEDREZ no es una primitiva de **Logo,** se definiría como TABLERO PIEZAS JGAR. Tampoco TABLERO e suna instrucción inicial, luego debemos describir TABLERO por ejemplo como MARCO CASILLAS, que a su vez serán especificados finalmente a partir de comandos primitivos. Para más detalles véase la referencia 3 de la bibliografía.

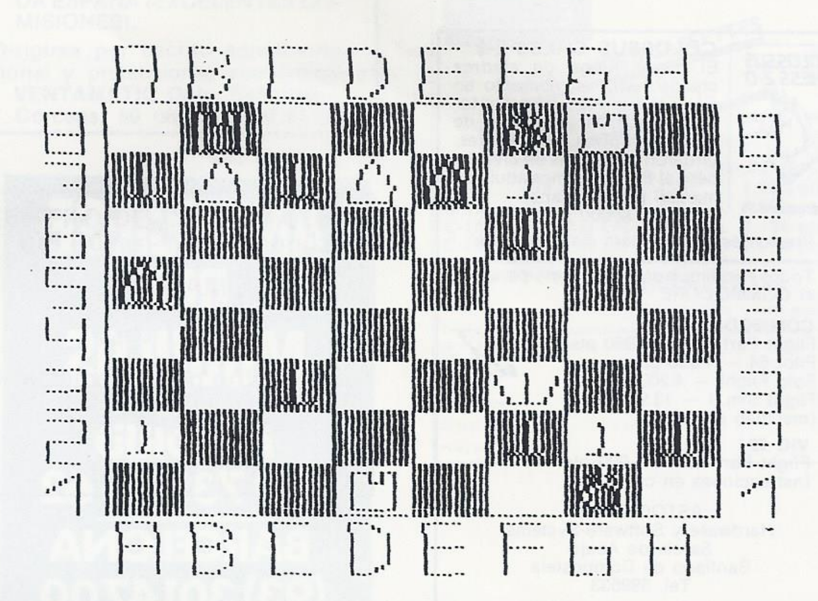


FIGURA 1.

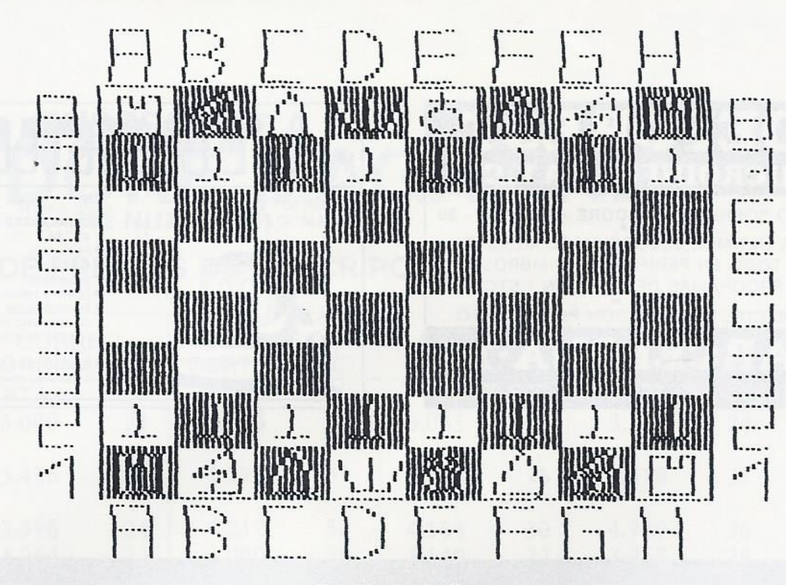


FIGURA 2.

José Miguel Aguirregabiria.
Profesor titular de la Escuela Universitaria de Profesorado de EGB de Bilbao.
Coordinador del equipo de EAO de Vizcaya.

BIBLIOGRAFIA BASICA

Lamentablemente existen aún pocos libros sobre **Logo** encstellano, por lo que se incluyen entre estas referencias textos en inglés y francés. Los títulos dedicados a **Apple** son plenamente válidos para el **CBM 64** sin apenas transformaciones (recuérdese que la relación entre los microprocesadores 6502 y 6510).

1. "DESAFIO A LA MENTE". Symour Papert. Ed. Galápago. Buenos Aires, 1981.
2. "ALAS PARA LA MENTE". Horacio Reggini. Ed. Galápago. Buenos Aires, 1981.
3. "LOGO: UN LENGUAJE DE PROGRAMACION". (En prensa). J. M. Aguirregabiria. Ed. Urmo. Bilbao, 1984.



SUSCRIBASE A
COMMODORE MAGAZINE
(12 NUMEROS)

TARIFA DE PRECIOS DE SUSCRIPCION

	CORREO ORDINARIO		CORREO CERTIFICADO		CORREO AEREO		CORREO AEREO-CERTIF.	
	PTAS.	$	PTAS.	$	PTAS.	$	PTAS.	$
ESPAÑA	3.000	21	3.273	23	3.055	22	3.333	24
EUROPA, MARRUECOS, TUNEZ, TURQUIA, ARGELIA Y CHIPRE	3.456	25	4.272	31	3.600	26	4.418	31
COSTA RICA, CUBA, CHILE, PARAGUAY Y REP. DOMINICANA	3.396	24	4.212	30	4.164	30	4.980	36
GIBRALTAR Y PORTUGAL	3.264	23	4.080	29	3.149	22	3.965	28
FILIPINAS	3.264	23	3.540	25	3.775	27	4.050	29
RESTO DEL MUNDO	3.456	25	4.272	31	4.224	30	5.040	36

CUPON DE PEDIDO

Recorte y envíe este cupón a: **COMMODORE MAGAZINE,** EDISA. Lopez de Hoyos, 141- MADRID - 28002

El importe lo abonaré: POR CHEQUE □ CONTRA REEMBOLSO □
CON TARJETA DE CREDITO □ American Express □ Visa □ Interbank □

Número de mi Tarjeta: ____________ Fecha de caducidad: ________

NOMBRE ____________

DIRECCION ____________

CIUDAD ____________ D.P. ____ PROVINCIA ____________

CALCULO FINANCIERO

Con este programa podemos saber cuanto tiempo necesitamos para hacernos ricos. Con él podemos hallar el interés que rinde un capital por los métodos de interés simple e interés compuesto, calculando además las tablas de amortización de capital (devolución de préstamos), para saber cuanto nos va a cobrar el banco por el préstamo que vamos a pedirle, o cuanto le debemos a cobrar a alguien que nos ha pedido una cantidad de dinero. Y todo con la ayuda del **Commodore 64.**

Para facilitar la operación, disponemos de unas máscaras que posibilita tres opciones:

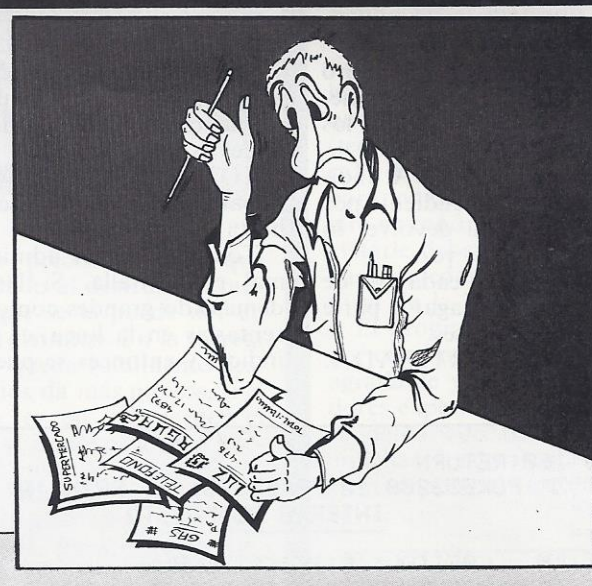
Opción 1: Cuadro de amortización de un capital: El programa pregunta el capital, que deberá tener una longitud máxima de siete dígitos en la parte entera para poder ser representados en pantalla todos los datos. La parte decimal no está limitada, pero por la misma razón sólo se representará un decimal en la tabla, aunque el programa los tiene en cuenta a todos.

A continuación pide el porcentaje y después de éste, el tiempo en que se va a amortizar el capital (o devolver el préstamo), no pudiendo ser superior a dos dígitos y teniendo en cuenta que el porcentaje ha de estar en concordancia con los períodos de tiempo en que se irá devolviendo. Es decir, si el capital a devolver se irá pagando mensualmente con un interés del 24

```
0 REM *** CALCULO FINANCIERO  CBM-64 ***
1 REM *** POR M. DE BLAS.  10/8/84   ***
10 POKE53280,14:POKE53281,6:PRINT"■";
15 PRINT"■█████";
20 PRINT"             CALCULO FINANCIERO
30 PRINT"             ──────────────────
40 PRINT"█        1 - CUADRO DE AMORTIZACION                  DE UN CAPITAL.
50 PRINT"█        2 - INTERES COMPUESTO.
60 PRINT"█        3 - INTERES SIMPLE.
70 PRINT"████              ▮OPCION: ?
80 GETX$:IFX$<>"1"ANDX$<>"2"ANDX$<>"3"THENGOTO80
90 X=VAL(X$):ONX GOTO 100,500,700
100 PRINT"■■":PRINT"             CUADRO DE AMORTIZACION"
101 PRINT"      ──────────────────────────────"
103 POKE53280,14:POKE53281,6:TC=17:AM=0:TT=0
105 INPUT"█   CAPITAL";C:CAP=C:CAP$=STR$(INT(CAP)):CC=C-INT(C)
110 IF LEN(CAP$) >8 THEN PRINT"█      SIETE DIGITOS ENTEROS MAXIMO":GOTO105
115 CC$=MID$(STR$(CC),2,2)
117 IFC=INT(C)THENCC$=""
120 INPUT"█   PORCENTAJE";R:R=R/100
130 INPUT"█   TIEMPO";T:T$=STR$(T)
140 IFLEN(T$)>3THENPRINT"█                DOS DIGITOS MAXIMO":GOTO130
145 POKE53280,12:POKE53281,11:PRINT"▮"
150 A=(C*R*(1+R)↑T)/((1+R)↑T-1)
155 N1=INT(A):N2=A-N1:N1$=STR$(N1):N2$=MID$(STR$(N2),2,2)
157 IF LEN(N2$)<2 THEN N2$="   "
158 GOSUB160:GOTO210
160 PRINT"■":PRINT" ANUALIDAD";N1$;N2$;"  CAPITAL";CAP$;CC$
170 PRINT"█──┬────────┬────────┬────────┬──────────";
180 PRINT"TI |CANTIDAD▮|INTERES  |CANTIDAD |TOTAL"
190 PRINT"EM |ADEUDADA▮|         |AMORTIZ. |AMORTIZ. ";
195 PRINT"──┼────────┼────────┼────────┼──────────";
200 RETURN
210 FORI=1TOT
211 I$=MID$(STR$(I),2)
212 IF LEN(I$)<2 THEN I$=" "+I$:GOTO212
```

por ciento anual, el porcentaje a introducir será 24/12 meses = dos por ciento, y si se va a devolver en un año, el tiempo son doce meses.

Durante la introducción de estos datos se puede volver al menú entrando un número negativo, indicando también el programa si la cifra es demasiado larga. Una vez introducidos estos datos, el programa indica la anualidad o cuota a pagar cada período de tiempo y el capital a devolver así como el cuadro de amortización en sí mismo que consta de las columnas:



```
213 PRINTI$;"|";
220 CAP=CAP-AM:C1=INT(CAP):C2=CAP-C1:C1$=MID$(STR$(C1),2):C2$=MID$(STR$(C2),2,2)
230 IF LEN(C1$)<7 THEN C1$=" "+C1$:GOTO230
233 IF C2<.01 THEN C2$=".0"
235 IF CAP=C1 THEN C2$="  "
240 PRINTC1$;C2$;"|";
250 IN=CAP*(1+R)-CAP:I1=INT(IN):I2=IN-I1
260 I1$=MID$(STR$(I1),2):I2$=MID$(STR$(I2),2,2):IFLEN(I1$)>6THEN390
270 IFLEN(I1$)<6THEN I1$=" "+I1$:GOTO270
273 IF I2<.01 THEN I2$=".0"
275 IF IN=I1 THEN I2$="  "
280 PRINTI1$;I2$;"|";
290 AM=A-IN:A1=INT(AM):A2=AM-A1:A1$=MID$(STR$(A1),2):A2$=MID$(STR$(A2),2,2)
300 IF LEN(A1$)>6 THEN 390
310 IF LEN(A1$)<6 THEN A1$=" "+A1$:GOTO310
312 IF A2<.01 THEN A2$=".0"
315 IF AM=A1 THEN A2$="  "
320 PRINTA1$;A2$;"|";
330 TT=TT+AM:T1=INT(TT):T2=TT-T1:T1$=MID$(STR$(T1),2):T2$=MID$(STR$(T2),2,2)
340 IF LEN(T1$)>7THEN390
350 IF LEN(T1$)<7 THEN T1$=" "+T1$:GOTO350
353 IF T2<.01 THEN T2$=".0"
355 IF TT=T1 THEN T2$="  "
360 PRINTT1$;T2$;
370 IFI=TCTHENTC=TC+17:GOSUB420
380 NEXT I
385 GOTO393
390 PRINT:PRINT"▓ SOBREPASADA CAPACIDAD DE REPRESENTACION
392 PRINT"▓                EN PANTALLA
393 PRINT"▓▓▓ PULSE ▓M▓ PARA MENU, O ▓T▓ PARA OTRA TABLA
395 GETX$:IFX$<>"M"ANDX$<>"T"THEN395
400 IFX$="M" THEN 10
410 GOTO100
420 PRINT"▓        PULSE UNA TECLA O ▓M▓ PARA MENU▓";
423 GETX$:IFX$=""THEN423
425 IF X$="M" THEN RUN
```

TIEM (tiempo): indica el número de orden del período de tiempo (meses, años...) a que hacen referencia los datos de la fila.

CANTIDAD ADEUDADA: indica el capital que queda pendiente por devolver por si interesa devolver lo que queda, de una sola vez.

INTERES: Indica, en cada una de las cuotas que se deben pagar la parte correspondiente a intereses.

CANTIDAD AMORTIZADA: Indica el importe correpondiente a amortización de capital (o capital devuelto) en cada una de las anualidades o cuotas.

TOTAL AMORTIZADO: Indica la parte del capital que se ha devuelto hasta ese momento.

Todas las cifras admiten un decimal en pantalla. Si las cifras son demasiado grandes como para representarlas en la línea, el programa lo indica, y entonces se puede volver al menú, o hacer otra tabla. En la pantalla caben hasta 17 plazos. Si hay más, se representarán los 17 siguientes pulsando una tecla, pudiéndose también volver al menú, pulsando "M". Cuando el proceso termina se puede volver al menú, o hacer otra tabla, pulsando "M" o "T" respectivamente. Todo esto lo va indicando el programa.

Opción 2: interés compuesto: Esta opción permite calcular el interés que rinde un capital mediante la fórmula de interés compuesto. Primero pregunta el capital, a continuación el porcentaje y por último el tiempo. Estos datos no están limitados por el programa como en el caso anterior. Si se introduce un dato negativo se puede volver al menú principal. Una vez introducidos los datos se muestra el interés que producirá el capital, así como el montante (capital más intereses). Pulsando cualquier tecla se puede hallar otro, o volver al menú pulsando "M".

Opción 3: interés simple: El programa primero pregunta cómo se va a medir el tiempo, pudiéndose optar por días, meses o años, pulsando "D", "M" o "A" respectivamente. A continuación se pide el capital, luego el porcentaje y por último el tiempo. Si se entra un número negativo se puede volver al menú. Estas cifras tampoco tienen límite, excepto la propia capacidad del ordenador. Una vez introducidos los datos se muestran los intereses y el montante, pudiéndose calcular otro pulsando cualquier tecla o pasar al menú pulsando "M".

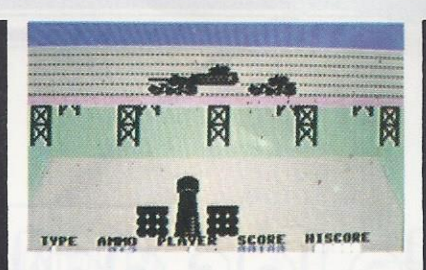
Comprobación: Para comprobar que se ha pasado bien el programa, un capital de 1.000.000 a un 3.75 por ciento de interés mensual, a devolver en 12 meses, en la fila 12 de la tabla debe de dar: cantidad adeudada: 101216.6 intereses: 3795.6; cantidad amortizada: 101216.6; total amortizado: 999999.9 (debido a los decimales). Anualidad: 105012.2

```
430 GOSUB 160:RETURN
500 PRINT"▪":POKE53280,12:POKE53281,11:PRINT"▪"
505 PRINT"                    INTERES COMPUESTO
506 PRINT"                    _________________
510 INPUT "▪   CAPITAL";C:IFC<0THENRUN
520 INPUT "▪   PORCENTAJE";R:IF R<0 THEN RUN
525 R=R/100
530 INPUT"▪   TIEMPO";T:IF T<0 THEN RUN
540 P=C*((1+R)↑T)
550 PRINT"▪_______________________________________"
560 IN=P-C:PRINT"▪    INTERESES:";
565 PRINTTAB(29-LEN(STR$(INT(IN))));IN
600 PRINT"▪    MONTANTE :";
605 PRINTTAB(29-LEN(STR$(INT(P))));P
610 PRINT"▪▪▪    PARA CONTINUAR PULSA UNA TECLA"
620 PRINT"▪    O PULSA ▮M■  PARA VOLVER AL MENU"
630 GETX$:IFX$=""THEN630
640 IFX$="M"THEN RUN
650 GOTO500
700 POKE53280,12:POKE53281,11:PRINT"▪"
710 PRINT"▪":PRINT"                    INTERES SIMPLE"
720 PRINT"                    ______________"
730 PRINT"▪  LA UNIDAD DE TIEMPO SERA:"
740 PRINT"▪ ▮▮▮▮D■ = DIAS;   ▮M■ = MESES;   ▮A■ = ANYOS"
750 GETX$:IFX$<>"D"ANDX$<>"M"ANDX$<>"A"THEN 750
760 IF X$="D" THEN K=360:K$="DIAS"
770 IF X$="M" THEN K=12:K$="MESES"
780 IF X$="A" THEN.K=1:K$="ANYOS"
790 INPUT"▪   CAPITAL";C:IF C<0 THEN RUN
800 INPUT"▪   PORCENTAJE";R:IF R<0 THEN RUN
810 PRINT"▪   N. DE ";K$;:INPUTT:IF T<0 THEN RUN
820 I=(C*R*T)/(100*K):M=C+I
830 PRINT"________________________________________"
840 PRINT"▪    INTERESES:";
845 PRINTTAB(29-LEN(STR$(INT(I))));I
850 PRINT"▪    MONTANTE :";
860 PRINTTAB(29-LEN(STR$(INT(M))));M
870 PRINT"▪▪    PARA CONTINUAR PULSA UNA TECLA"
880 PRINT"▪    O PULSA ▮M■  PARA VOLVER AL MENU";
890 GETX$:IFX$=""THEN890
900 IF X$="M" THEN RUN
1000 GOTO700
```

Mariano de Blas

PROGRAM: 3D TANX
TIPO: JUEGO
DISTRIBUIDOR: ABC SOFT
FORMATO: CASSETTE
COMPUTADOR:
COMMODORE 64

Aunque el nombre del juego nos induzca a pensar en un aparatoso juego en tres dimensiones, como el BATLEXONE de **ATARI**, que podemos ver en muchos bares, la realidad es bastante distinta. El juego nos coloca en la torreta de mando de un tanque que debe destruir a todos los blindados enemigos que pasen por un puente situado enfrente nuestro, éstos van circulando de un lado a otro sin pararse por varios carriles. Si logramos destruir a un tanque, todos los que le sigan por ese mismo carril se detendrán, hasta que uno de ellos vuele al destruido de un cañonazo. Durante ese tiempo la fila se queda parada y podemos hacer fuego sobre ellos



con gran precisión. Hay que tener cuidado al destruir un tanque, ya que se le puede dar en dos sitios distintos. En la torreta el tanque queda totalmente destruido y nos da más puntos, si el cañonaño es en el cuerpo, el blindado se inmoviliza pero puede seguir usando el cañón y disparará sobre nosotros, por lo que deberemos dispararle por segunda vez. Para disparar podemos controlar el giro y la inclinación del cañón dentro de un amplio campo, por lo que hay que afinar la puntería bastante.

El planteamiento es sencillo y no tiene las complicaciones de otros juegos que están apareciendo en el mercado. El juego llega a parecerse demasiado a un tiro al blanco en elque los patitos han sido sustituidos por tanques que se deslizan a distintos niveles (teóricamente más o menos cerca). A veces (demasiado pocas para nuestro gusto) un tanque gira su cañón y nos dispara, para evitarle debemos girar la torreta hasta quedar fuera de la trayectoria de la bala.

El programa está bien realizado, con buenos gráficos, manejo agradable y sencillo. A los jugadores experimentados les parecerá demasiado tonto, pero para aquellas personas que se inician en el mundo del videojuego.

PUNTUACION:
ADICCION: 6
PRESENTACION: 6
GRAFICOS: 7 ACCION: 5

PROGRAMA:
MOON BUGGY
TIPO: JUEGO
DISTRIBUIDOR:
INDESCOMP
FORMATO: CASSETTE
COMPUTADOR:
COMMODORE 64 CON JOYSTICK

Este programa nos coloca en la superficie de la Luna conduciendo un avanzado "buggy" que puede acelerar, saltar y hacer fuego de frente y hacia arriba. La cinta incorpora un cargador turbo que realiza la carga con gran rapidez, cosa muy agradable comparada con la lentitud habitual del **Commodore** en estos casos.

Una vez puesto en marcha el programa se nos muestra un corte en sección del terreno y, por encima, nuestro vehículo que se va deslizando a velocidad constante. Haciendo uso de los



controles izquierdo y derecho del *joystick* podemos acelerarlo y frenarlo (aunque no pararlo del todo), si empujamos el mando hacia arriba (o hacia el frente, según se mire) el vehículo saltará un trecho. Por último, el botón del *joystick* dispara dos cohetes, uno situado en la parte superior hacia arriba y otro hacia el frente.

Durante el viaje van apareciendo peligros de la más diversa índole, al principio son pequeños cráteres que hay que saltar para evitar estrellarnos, luego vendrá un platillo volante, que debemos destruir con el cohete superior. Posteriormente aparecen enemigos delante a los que hay que lanzar más cohetes, además de gran cantidad de minas repartidas por el terreno. Para evitar éstas últimas, existen dos sistemas, si la mina sobresale mucho del terreno debemos dispararle antes de llegar a ella, si por el contrario está enterrada, debemos saltar por encima de ella como si fuese un cráter.

El programa tiene una presentación soberbia, los gráficos tienen una calidad extraordinaria y la animación está muy bien conseguida. El juego ensi es algo difícil de dominar y se necesita una gran paciencia al principio, hasta que se llega a controlarlo totalmente.

PUNTUACION:
ADICCION: 7
PRESENTACION: 8
GRAFICOS: 8 ACCION: 9

Programas

El piloto de las galaxias

CBM 64

Este programa para el C-64 convertirá la pantalla de su televisor en el cristal de la cabina de una nave intergaláctica. Sobre dicho cristal aparece un punto de mira: es el del láser de la nave. Tu misión es moverte por el espacio destruyendo el mayor número de naves enemigas posible, dentro de los dos minutos de tiempo de que dispones. Para ello debes desplazar tu nave hasta tener a las naves enemigas bien enfiladas en tu punto de mira, entonces dispara. Hay muchas naves enemigas que aparecerán en cualquier momento, cuando menos te lo esperes.

Para controlar tu nave y disparar el láser debes utilizar las siguientes teclas del teclado:

W=Arriba.
X=Abajo.
A=Izquierda.
D=Derecha.
S=Dispara el láser.

El programa, para conseguir mayor velocidad, utiliza toda una serie de rutinas en lenguaje máquina, que se introducen en la memoria del ordenador a partir de las sentencias data; por ello es de la mayor importancia no equivocarse con estas DATA, y en cualquier caso, asegurarse de que el programa está guardando en cinta antes de escribir RUN, ya que si no, podríamos quedarnos colgados teniendo que apagar el ordenador y sin ninguna copia del programa que hemos tecleado.


```
10 REM ********************************
15 REM * EL PILOTO DE LAS GALAXIAS *
20 REM *                           *
25 REM *    COMMODORE MAGAZINE     *
30 REM ********************************
35 :
40 POKE55,255:POKE56,47:V=53248:POKEV+32,12:POKEV+33,0:POKEV+38,5:POKE54296,15
45 FORI=0TO638:READA:POKE1024*12+I,A:NEXT
50 POKE2040,192:POKE2041,193:POKE2042,194:POKEV+21,1:POKEV+39,5:POKEV+23,1
55 POKEV+29,9:POKEV,148:POKEV+1,108:POKEV+28,6:POKEV+40,2:POKEV+41,2:POKEV+37,8
60 KB=197:PRINT"♥"
65 FORI=0TO151:READA:POKE13*1024+I,A:NEXT
70 D=13*1024:LE=D+52:RI=D+102:FORI=55296TO56295:POKEI,1:NEXT
75 FORI=1TO50:POKE1024+RND(1)*1000,46:NEXT
80 AL=0:TI$="000000":SC=0
85 K=PEEK(KB):A=PEEK(V+30):POKEV+39,5:IFA=9THENPOKEV+39,2
90 IFK<>13THEN160
95 POKE54276,0:POKE54273,72:POKE54272,169:POKE54277,143:POKE54276,129
100 POKEV+2,60:POKEV+3,220:POKEV+5,220
105 POKEV+4,4:POKEV+16,PEEK(V+16)OR4:POKEV+21,PEEK(V+21)OR6
110 FORI=1TO32:POKEV+2,PEEK(V+2)+3:POKEV+3,PEEK(V+3)-3:POKEV+5,PEEK(V+3)
115 A=PEEK(V+4)-3:IFA<0THENA=255:POKEV+16,PEEK(V+16)-4
120 POKEV+4,A
125 A=PEEK(V+30):NEXT:POKEV+21,PEEK(V+21)-6:B=AAND2:IFB=0THENB=AAND4
130 IFB=0THEN145
135 IFAAND8THEN390
140 POKE54276,0:POKE54273,17:POKE54277,7:POKE54276,129
145 POKEV+2,PEEK(V)+10:POKEV+3,PEEK(V+1)+10:POKE2041,201:POKEV+21,PEEK(V+21)OR2
```




```
150 FORI=1TO50:NEXT:POKEV+21,PEEK(V+21)-2
155 POKE2041,193
160 IFAL>0THEN230
165 IFRND(1)>.1THEN285
170 IFPEEK(V+16)ANDATHENPOKEV+16,PEEK(V+16)-8
175 POKEV+6,RND(1)*256:IFRND(1)>.75THENPOKEV+6,RND(1)*63:POKEV+16,PEEK(V+16)OR8
180 POKEV+7,RND(1)*200:Z=INT(RND(1)*5):POKEV+42,4
185 DZ=INT(RND(1)*5-2):DY=INT(RND(1)*5-2):DX=INT(RND(1)*5-2)
190 IF(DX=0ANDDY=0ANDDZ=0)OR(DX=0ANDDY=0ANDDZ>0)THEN185
195 IFZ=0THEN225
200 IFPEEK(V+16)AND8THENPOKEV+16,PEEK(V+16)-8
205 IFRND(1)>.5THEN220
210 POKEV+6,2:IFRND(1)>.5THENPOKEV+6,86:POKEV+16,PEEK(V+16)OR8
215 GOTO225
220 POKEV+7,31:IFRND(1)>.5THENPOKEV+7,246
225 POKE2043,195+Z:POKEV+21,PEEK(V+21)OR8:AL=1
230 A=PEEK(V+7)+DY*2:IFA<31ORA>246THENAL=0:POKEV+21,PEEK(V+21)-8:GOTO285
```

Viene de la página anterior

```
235 POKEV+7,A:A=PEEK(V+6)+DX*2:B=PEEK(V+16)AND8
240 IF(A<2ANDB=0)OR(A>86ANDB)THENAL=0:POKEV+21,PEEK(V+21)-8:GOTO285
245 IFA<0THENA=255:POKEV+16,PEEK(V+16)-8
250 IFA>255THENA=0:POKEV+16,PEEK(V+16)OR8
255 POKEV+6,A:Z=Z+DZ/20:IFZ<0THENAL=0:POKEV+21,PEEK(V+21)-8:GOTO285
260 IFZ>4.9THENZ=4.9:DZ=-INT(RND(1)*3)
265 POKE2043,195+Z
270 IFRND(1)<.9THEN285
275 DX=INT(RND(1)*5-2):DY=INT(RND(1)*5-2):IFRND(1)>.7THENDZ=INT(RND(1)*5-2)
280 IFDX=0ANDDY=0ANDDZ=0THEN275
285 IFK<>9ANDK<>18ANDK<>23ANDK<>10THEN360
290 IFAL=0THEN335
295 A=PEEK(V+7)-(K=9)*8+(K=23)*8
300 IFA<31 ORA>246THENAL=0:POKEV+21,PEEK(V+21)-8:GOTO85
305 POKEV+7,A
310 B=PEEK(V+16)AND8:A=PEEK(V+6)-(K=10)*8+(K=18)*8
315 IF(A<2ANDB=0)OR(A>86ANDB)THENAL=0:POKEV+21,PEEK(V+21)-8:GOTO285
320 IFA<0THENA=255:POKEV+16,PEEK(V+16)-8
325 IFA>255THENA=0:POKEV+16,PEEK(V+16)OR8
330 POKEV+6,A
335 IFK=9THENSYSD:POKE1024+RND(1)*40,46:GOTO360
340 IFK=10THENSYSRI:POKE1024+INT(RND(1)*25)*40,46:GOTO360
345 IFK=18THENSYSLE:POKE1063+INT(RND(1)*25)*40,46:GOTO360
350 PRINT"[illegible]"
355 POKE56295,1:POKE1984+RND(1)*40,46
360 IFTI$<"000200"THEN85
365 PRINT"[illegible]SE ACABO EL TIEMPO [illegible]":PRINT"[illegible]PUNTUACION";SC:POKEV+27,255
370 PRINT"[illegible]TE GUASTARIA JUGAR OTRA VEZ? S/N"
375 GETA$:IFA$="S"THENPOKEV+21,1:PRINT"[illegible]":POKEV+27,0:GOTO70
380 IFA$<>"N"THEN375
385 END
390 POKEV+21,9:AL=0:POKEV+2,PEEK(V)+10:POKEV+3,PEEK(V+1)+10
395 POKE54276,0:POKE54273,8:POKE54272,47:POKE54277,76:POKE54276,129
400 POKEV+4,PEEK(V):POKEV+5,PEEK(V+1):POKEV+23,PEEK(V+23)OR4
405 POKEV+29,PEEK(V+29)OR4:POKE2041,200:POKE2042,200
410 POKEV+21,PEEK(V+21)OR6:FORA=200TO201:POKE2042,A:FORB=200TO201.9STEP.05
415 POKE54276,0:POKE54276,129:POKE2041,B:NEXTB,A:POKEV+23,1:POKEV+29,9
420 POKEV+21,1:POKE2041,193:POKE2042,194:SC=SC+(5-INT(Z))*10:GOTO160
425 DATA0,0,0,0,0,0,0,0,0,28,0,0,255,128,7,128,240,12,0,24,16,8,4,32,8,2
430 DATA32,0,2,0,99,0,32,0,2,32,8,2,16,8,4,12,0,24,7,128,240,0,255,128,0,28,0
435 DATA0,0,0,0,0,0,0,0,99
440 DATA0,0,0,0,0,0,2,128,0,2,128,0,41,128,0,41,128,0,150,0,0,150,0,2,86,0
445 DATA2,86,0,2,88,0,2,88,0,9,160,0,9,160,0,38,0,0,38,0,0,40,0,0,40,0,0
450 DATA128,0,0,128,0,0,0,0,0,99
455 DATA0,0,0,0,2,128,0,2,128,0,2,104,0,2,104,0,0,150,0,0,150,0
460 DATA0,149,128,0,149,128,0,37,128,0,37,128,0,10,96,0,10,96,0,0,152,0,0,152
465 DATA0,0,40,0,0,40,0,0,2,0,0,2,0,0,0,99
470 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,8,0,0,62,0,0,0,0
475 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,99
480 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,8,0,0,28,0,0,255,128
485 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,99
490 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,34,0,0,28,0,0,127,0,3,255,224
```

```
495 DATA0,201,128,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
500 DATA99,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,99,0,0,28,0,0,62,0,0,247,128
505 DATA7,255,240,31,255,252,1,201,192,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
510 DATA0,0,0,0,0,0,0,99
515 DATA0,0,0,0,0,0,0,0,0,0,0,0,128,128,0,99,0,0,28,0,0,127,0,1,227,192,15
520 DATA255,248,127,255,255,7,156,240,2,0,32,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
525 DATA0,0,0,0,0,0,0,99
530 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,32,0,0,186,128,10,247,128,47,123,128
535 DATA181,151,224,173,85,120,11,95,248,2,222,160,2,222,0,2,222,0,0,190,0
540 DATA0,40,0,0,0,0,0,0,0,0,0,0,0,0,99
545 DATA0,0,0,0,0,0,2,128,0,138,232,2,239,120,11,117,224,45,119,168,47,87
550 DATA254,10,213,94,11,85,248,11,87,160,45,221,128,181,221,224,183,223,120
555 DATA46,187,94,8,34,222,0,0,184,0,0,32,0,0,0,0,0,0,0,0
560 DATA162,192,189,255,6,157,39,7,202,208,247,202,189,255,5,157,39,6,202,208
565 DATA247,202,189,255,4,157,39,5,202,208,247,202,189,255,3,157,39,4,202,208
570 DATA247,169,32,157,0,4,232,224,40,208,248,96,
575 DATA169,0,133,20,169,4,133,21,160,1,177,20,136,145,20,200,200,192,40
580 DATA208,245,136,169,32,145,20,169,40,24,101,20,133,20,169,0,101,21,133,21
585 DATA201,7,208,221,165,20,201,232,208,215,96
590 DATA169,0,133,20,169,4,133,21,160,39,177,20,200,145,20,136,136,192,255
595 DATA208,245,200,169,32,145,20,169,40,24,101,20,133,20,169,0,101,21,133,21
600 DATA201,7,208,221,165,20,201,232,208,215,96
```

Programas

Cucaracha cosmica

C-64



En este programa una serie de insectos cósmicos intentan llegar desde lo alto de la pantalla hasta el suelo, para una vez allí, y convertidos en cangrejos, acorralar al jugador y acabar con una de sus vidas. Para defenderse se dispone de dos tipos de armas, por un lado se pueden utilizar redes que atrapan a los insectos y que proporcionan una puntuación doble, por otro lado se dispone de paquetes de energía que destruyen a los insectos, pero que proporcionan menos puntos que el empleo de las redes. Si la pantalla se vuelve de color amarillo hay que prepararse para ascender hasta la parte superior de la misma y seguir jugando en una nueva zona de juego.

Las criaturas cósmicas son las siguientes:

Escorpiones: Son de color verde y su valor pueden ser de 20 puntos o de 30 puntos según que vayan o no dejando una traza al acercarse.

Cortadores: Son de color azul y tienen la propiedad de poder liberar a las criaturas atrapadas por una red. Su valor es de 40 puntos y no se les puede atrapar con redes, sino que hay que matarlos.

Cangrejos cósmicos: Es en lo que se convierte cualquier criatura que consiga llegar a la línea inferior de la pantalla. La única forma de acabar con ellos es utilizando la "gran explosión" (barra de espacios) lo que consume **25 puntos.** Si se dispone de suficientes puntos la pantalla permanece de color gris. Además, la primera vez que se utiliza esta terrible arma, acaba no sólo con los cangrejos, sino con toda forma de vida que hubiera sobre la pantalla.

El control del programa se lleva a cabo mediante las teclas:

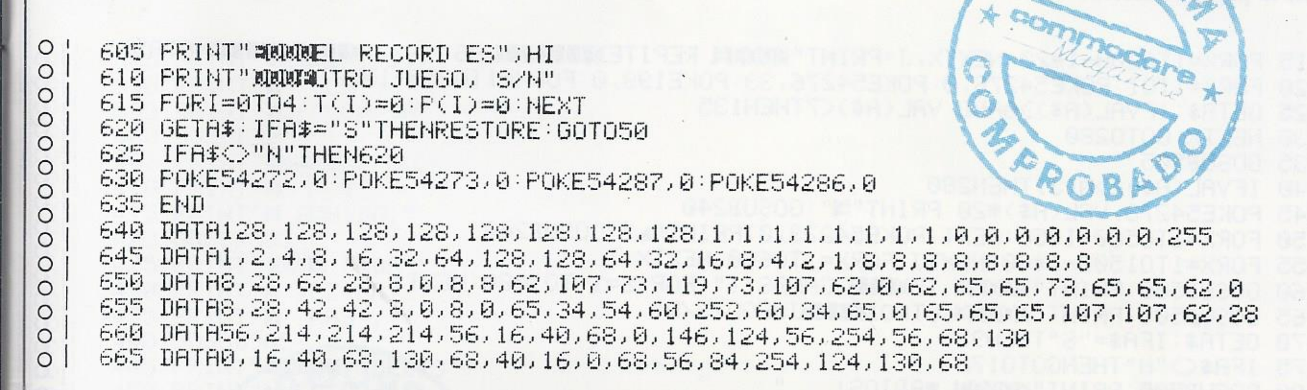
Z=Izquierda.
C=Derecha.
>=Disparo de redes.
<=Disparo de paquetes de energía.
Espacio=gran explosión.

```
10 REM ************************
15 REM * CUCARACHA  COSMICA *
20 REM *                    *
25 REM * COMMODORE MAGAZINE *
30 REM ************************
35 POKE55,255:POKE56,47:V=53248:POKEV+33,0:POKEV+32,0:POKE54296,15
40 DIMP(4),T(4):FORI=0TO4:T(I)=0:NEXT:CO=54272:KB=197:LS=1024:CA=0
45 PC=11:L=12:R=20:HI=0:FORI=12544TO12552:POKEI,0:NEXT
50 FORI=12288TO12415:READA:POKEI,A:NEXT:POKEV+24,28
55 SC=0:LI=3:P=2004:NE=20:N=-1:S=-1:SZ=0:RE=0:BO=0
60 PRINT"▯█ ";:FORI=1TO24:PRINT"██@";:NEXT:PRINT"███":FORI=1TO23:PRINT"██A";
65 NEXT:POKE56295,1:POKE2023,1:PRINT"████████████████████████████████";
```

```
70 PRINT"C██C██C██████BBBBC                    DBBBB███████D███D███D█C.C.C.C.C.C";
75 PRINT"C.C.C.C.C.C.C.C";:TI$="000000":POKE54278,32:POKE54277,0:POKE54276,0
80 POKE54276,17
85 FORI=0TO4:K=PEEK(KB):IFK<>LANDK<>RTHEN100
90 POKEP,32
95 P=P+(K=L)-(K=R):IFPEEK(P)<>32THENP=P-(K=L)+(K=R)
100 POKEP+CO,7:POKEP,PC
105 IFT(I)>0THEN125
110 IFRND(1)<.9THEN330
115 P(I)=INT(LS+RND(1)*40)
120 T(I)=INT(RND(1)*2+1):IFRND(1)<CA/6THENT(I)=4
125 ONT(I)GOTO130,165,205,235,285,305,305,305
130 CL=2:CH=12:D=INT(RND(1)*3-1)+40:PO=P(I)+D:PE=PEEK(PO):IFPE=PCTHEN565
135 IFPE=5THENPE=32
140 IFPE=7THENT(I)=6:POKEP(I),32:P(I)=PO:CA=CA+1:SC=SC+20:GOTO305
145 IFPE<>32ANDRND(1)>.5THEND=INT(RND(1)*3-1):PO=P(I)+D:PE=PEEK(PO):GOTO140
150 IFPE<>32THEND=0:GOTO320
155 IFPO>1983THEN230
160 GOTO320
165 CL=5:CH=13:D=INT(RND(1)*3-1)+40:PO=P(I)+D:PE=PEEK(PO):IFPE=PCTHEN565
170 IFPE=5THENPE=32
175 IFPE=7THENT(I)=7:POKEP(I),32:CA=CA+1:P(I)=PO:SC=SC+40:GOTO305
180 IFPE<>32ANDRND(1)>.5THEND=INT(RND(1)*3-1):PO=P(I)+D:PE=PEEK(PO):GOTO175
185 IFPE<>32THEND=0:GOTO320
190 IFPO>1983THEN230
195 IFRND(1)>.9ANDD>0THENT(I)=3
200 GOTO320
205 POKEP(I),5:CL=5:CH=13:D=40:PO=P(I)+D:PE=PEEK(PO):IFPE=PCTHEN565
210 IFPE=7THENT(I)=8:POKEP(I),32:P(I)=PO:CA=CA+1:SC=SC+60:GOTO305
215 IFPE=5THENPE=32
220 IFPE<>32THENT(I)=0:GOTO330
225 IFPO<1984THENP(I)=P(I)+D:GOTO325
230 T(I)=5:POKEP(I),32:P(I)=P(I)+D:POKEP(I)+C0,10:POKEP(I),15:GOTO330
235 CL=14:CH=14:D=INT(RND(1)*3-1)+40:PO=P(I)+D:PE=PEEK(PO):IFPE=PCTHEN565
240 IFPE=5THENPE=32
245 IFPE<>8THEN265
250 FORJ=0TO4:IFP(J)=POANDT(J)>5THENT(J)=T(J)-5:D=0:J=4:CA=CA-1:NE=NE+1
255 NEXTJ:GOTO320
260 IFPE<>32ANDRND(1)>.5THEND=INT(RND(1)*3-1):PO=P(I)+D:PE=PEEK(PO):GOTO240
265 IFPE<>32ANDRND(1)>.5THEND=INT(RND(1)*3-1):PO=P(I)+D:PE=PEEK(PO)
270 IFPE<>32THEND=0:GOTO320
275 IFPO>1983THENPOKEP(I),32:T(I)=0:GOTO330
280 GOTO320
285 CL=10:CH=15:D=INT(RND(1)*3-1):IFRND(1)>.1THEND=(P(I)>P)-(P(I)<P)
290 PO=P(I)+D:PE=PEEK(PO):IFPE=PCTHEN565
295 IFPE<>32THEND=0
300 GOTO320
305 CL=INT(RND(1)*15+1):CH=8:D=0
310 IFCA=5 THENPOKEP(J),32:T(J)=0:CA=CA-1:NE=NE+1:GOTO330
315 GOTO325
320 POKEP(I),32:P(I)=P(I)+D
325 POKEP(I)+CO,CL:POKEP(I),CH
330 IFK<>44ORN>-1ORNE=0THEN345
```

Viene de la página anterior

```
335 N=P:NE=NE-1:POKE54290,0:POKE54291,71:POKE54287,34:POKE54286,75
340 POKE54290,17
345 IFK<>47ORS>-1THEN355
350 S=P:POKE54290,0:POKE54291,71:POKE54287,72:POKE54286,169:POKE54290,33
355 IFN=-1THEN380
360 POKEN,32:N=N-40:IFN<LSTHENN=-1:GOTO380
365 IFRND(1)>.95 THENN=N+40:POKEN+CO,4:POKEN,7:N=-1:GOTO380
370 IFPEEK(N)=32 THENPOKECO+N,4:POKEN,6:GOTO380
375 N=N+40:POKEN+CO,4:POKEN,7:N=-1
380 IFS=-1THEN450
385 POKES,32:S=S-40:IFS<LSTHENS=-1:GOTO450
390 PE=PEEK(S):C=PEEK(S+CO):POKES+CO,8:POKES,9:IFPE=32THEN450
395 POKE54290,0:POKE54287,8:POKE54286,37:POKE54291,36:POKE54290,129
400 POKES+CO,7:POKES,10:FORJ=0TO4:IFT(J)=0ORP(J)<>STHEN425
405 IFT(J)>5THENT(J)=T(J)-5:SC=SC+5:J=100:CA=CA-1:NE=NE+1:GOTO425
410 SC=SC+T(J)*10:IFRND(1)<.7THENT(J)=0:J=100:GOTO425
415 FORJ1=0TO4:IFT(J1)=0THENT(J1)=T(J):P(J1)=P(J):J1=100:J=100
420 NEXTJ1:IFJ1<99THENT(J)=0:J=100
425 NEXTJ:IFJ>5THENPOKES,32:GOTO445
430 IFPE=7THENNE=NE+1:PE=32
435 IFPE=5THENPE=32
440 POKES+CO,C:POKES,PE
445 S=-1
450 POKE54273,RND(1)*50:POKE54272,RND(1)*256:IFK=60THEN530
455 RE=0:IFSC<25THENPOKEV+32,0
460 IFTI$>"000250"THENPOKEV+32,7:GOTO470
465 IFSC>=25THENPOKEV+32,11
470 IFBO=0ANDSC>1000THENSZ=0:BO=1
475 NEXTI:IFTI$<"000300"THEN85
480 IFS>-1THENPOKES,32:S=-1
485 IFN>-1THENPOKEN,32:N=-1
490 FORI=0TO4:IFT(I)>0THENPOKEP(I),32:T(I)=0
495 NEXT:NE=20:RE=0:SZ=0:CA=0
500 POKEV+33,7:FORI=1TO1000:NEXT:POKEV+33,0
505 K=PEEK(KB):POKEP,32:P=P-40+(K=L)-(K=R)
510 PE=PEEK(P):POKEP+CO,7:POKEP,PC:IFP<LSTHENP=2004:SC=SC+100:GOTO60
515 FORI=1TO50:NEXT
520 IFPE=32THEN505
525 LI=0:I=0:P(I)=P:GOTO565
530 IFSC<25ORRETHEN475
535 POKEV+33,13:SC=SC-25:FORJ=0TO4:IFT(J)=0THEN560
540 POKE54290,0:POKE54287,72:POKE54286,169:POKE54291,17:POKE54290,129
545 IFSZ=1ANDTJ<>5THEN560
550 IFT(J)>5THENCA=CA-1:NE=NE+1
555 POKEP(J),32:T(J)=0
560 NEXTJ:SZ=1:POKEV+33,0:RE=1:GOTO475
565 POKEP(I),32:POKEV+33,11:FORJ=1TO1000:NEXT
570 POKE54290,0:POKE54287,4:POKE54286,73:POKE54291,240:POKE54290,129
575 POKEV+33,12:FORJ=1TO1000:NEXT
580 POKEV+33,15:FORJ=1TO1000:NEXT
585 POKEV+33,0:FORJ=1TO1000:NEXT
590 T(I)=0:SZ=0:LI=LI-1:IFLI>0THEN450
595 POKEV+24,21:PRINT"[illegible]FIN DE JUEGO":PRINT"[illegible]PUNTUACION";SC
600 IFSC>HITHENHI=SC
```



```
605 PRINT"EL RECORD ES";HI
610 PRINT"OTRO JUEGO, S/N"
615 FORI=0TO4:T(I)=0:P(I)=0:NEXT
620 GETA$:IFA$="S"THENRESTORE:GOTO50
625 IFA$<>"N"THEN620
630 POKE54272,0:POKE54273,0:POKE54287,0:POKE54286,0
635 END
640 DATA128,128,128,128,128,128,128,128,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,255
645 DATA1,2,4,8,16,32,64,128,128,64,32,16,8,4,2,1,8,8,8,8,8,8,8,8
650 DATA8,28,62,28,8,0,8,8,62,107,73,119,73,107,62,0,62,65,65,73,65,65,62,0
655 DATA8,28,42,42,8,0,8,0,65,34,54,60,252,60,34,65,0,65,65,65,107,107,62,28
660 DATA56,214,214,214,56,16,40,68,0,146,124,56,254,56,68,130
665 DATA0,16,40,68,130,68,40,16,0,68,56,84,254,124,130,68
```

Simon-64

C - 64

Simón es un muñequito que te propone jugar con él. El juego consiste en repetir una secuencia de tonos y colores que él te irá marcando. La secuencia comienza con un solo color, después se repite el primer color y se añade otro nuevo, luego son tres colores, y así sucesivamente. Cada una de las secuencias puede estar constituida por hasta seis colores diferentes, repetidos o no. Cada vez que aciertes a repetir la secuencia correctamente, Simón le presentará de nuevo, a mayor velocidad, y añadiendo un color más. Para repetir la secuencia hay que utilizar las teclas del 1 al 6 correspondientes a cada uno de los seis colores. Si consigues recordar y repetir hasta ocho notas seguidas, ganarás el juego, pero ya verás cómo no es nada sencillo. El juego constituye una prueba a la memoria y a la velocidad de reflejos, ya que sólo dispones de unos pocos segundos para reaccionar.

Las variables empleadas por el programa son:
V=Puntero para el manejo de Sprites.
N(X)=Matriz de las notas tocadas.
A=Tecla pulsada por el jugador.
I=Indicador de la última tecla pulsada.

```
10 REM ***********************
15 REM *      S I M O N      *
20 REM *                     *
25 REM * COMMODORE MAGAZINE *
30 REM ***********************
35 :
40 PRINT"":POKE53281,1:POKE53280,4:POKE54296,15:POKE54276,0:POKE54276,17
45 POKE54277,0:POKE54278,240::POKE54273,0:V=53248:POKEV+21,3:POKE2040,13
50 POKE2041,14:POKEV+39,0:POKEV+40,5:POKEV+28,2:POKEV+38,0:FORI=0TO126:READQ
55 POKE832+I,Q:NEXT:POKEV,140:POKEV+2,140:POKEV+1,61:POKEV+3,82
60 PRINT"  _______________"
65 PRINT" |              L_"
70 PRINT" |              /"
75 PRINT" |              |"
80 PRINT" ________________"
85 PRINT" ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦":GOSUB 190
90 GOSUB335:DIMN(8)
95 FORI=1TO8:N(I)=INT(RND(1)*6)+1:FORJ=1TOI:POKE54276,17
100 POKE54273,N(J)*20:PRINT"":GOSUB240
105 FORX=1TO400-I*30:NEXT:POKE54273,0
110 PRINT"":GOSUB240
```

Programas

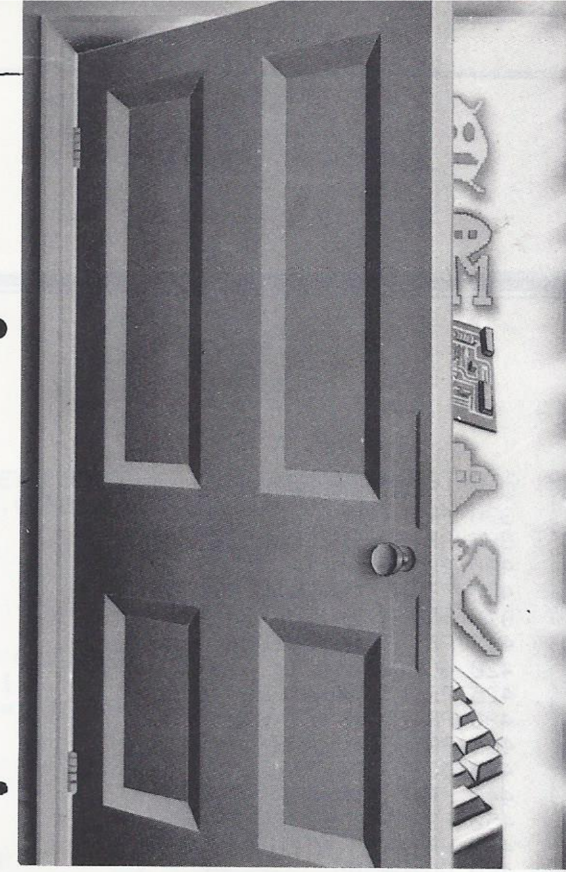
Viene de la página anterior

```
115 FORX=1TO200-I*20:NEXTX,J:PRINT"SQQ▮ REPITEQ▮▮▮▮▮▮▮LAS      Q▮▮▮▮▮▮▮▮▮NOTAS!"
120 FORJ=1TOI:POKE54276,0:POKE54276,33:POKE198,0:FORX=1TO250-I*20
125 GETA$:IFVAL(A$)>0AND VAL(A$)<7THEN135
130 NEXTX:GOTO280
135 GOSUB505
140 IFVAL(A$)<>N(J)THEN280
145 POKE54273,VAL(A$)*20:PRINT"▮":GOSUB240
150 FORX=1TO500-I*50:NEXT:POKE54273,0:PRINT"◣":GOSUB240
155 FORX=1TO1500-I*80:NEXTJ:FORX=1TO600:NEXTX,I
160 GOSUB505:PRINT"SQQ▮ TUQ▮▮▮ GANAS!!":FOR S=1 TO1000:NEXT
165 GOSUB505:PRINT"SQQ▮ OTROQ▮▮▮▮JUEGO?S/N"
170 GETA$:IFA$="S"THEN95
175 IFA$<>"N"THENGOTO170
180 GOSUB505:PRINT"SQQQ▮ ▮ADIOS!    "
185 END
190 PRINT"SQQQQQQQQQQ▮▮▮◤    ◥▮▮▮▮▮▮▮▮▮▮▮▮◤    ◥"
195 PRINT"▮▮▮▮  1   ▮▮▮▮▮▮▮▮▮▮▮▮▮  4   "
200 PRINT"▮▮▮◥▮    ◤▮▮▮▮▮▮▮▮▮▮▮▮◥▮    ◤"
205 PRINT"QQ▮▮▮◤    ◥▮▮▮▮▮▮▮▮▮▮▮▮◤    ◥"
210 PRINT"▮▮▮▮  2   ▮▮▮▮▮▮▮▮▮▮▮▮▮  5   "
215 PRINT"▮▮▮◥▮    ◤▮▮▮▮▮▮▮▮▮▮▮▮◥▮    ◤"
220 PRINT"QQ▮▮▮◤    ◥▮▮▮▮▮▮▮▮▮▮▮▮◤    ◥"
225 PRINT"▮▮▮▮  3   ▮▮▮▮▮▮▮▮▮▮▮▮▮  6   "
230 PRINT"▮▮▮◥▮    ◤▮▮▮▮▮▮▮▮▮▮▮▮◥▮    ◤"
235 RETURN
240 PRINT"SQQQQQQQQQQ▮▮▮";:IFN(J)>3THENPRINT"▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮";
245 IFN(J)=1THENPRINT"▮◤    ◥Q▮▮▮▮▮ 1  Q▮▮▮▮▮▮◥▮    ◤"
250 IFN(J)=2THENPRINT"QQQQQ▮◤    ◥Q▮▮▮▮▮ 2  Q▮▮▮▮▮▮◥▮    ◤"
255 IFN(J)=3THENPRINT"QQQQQQQQQQ▮◤    ◥Q▮▮▮▮▮ 3  Q▮▮▮▮▮▮◥▮    ◤"
260 IFN(J)=4THENPRINT"▮◤    ◥Q▮▮▮▮▮ 4  Q▮▮▮▮▮▮◥▮    ◤"
265 IFN(J)=5THENPRINT"QQQQQ▮◤    ◥Q▮▮▮▮▮ 5  Q▮▮▮▮▮▮◥▮    ◤"
270 IFN(J)=6THENPRINT"QQQQQQQQQQ▮◤    ◥Q▮▮▮▮▮ 6  Q▮▮▮▮▮▮◥▮    ◤"
275 RETURN
280 POKE54276,0:POKE54276,33
285 FORI=20TO1STEP-.2:POKE54273,I:NEXT
290 POKE54273,0
295 GOSUB505
300 PRINT"SQQ▮LASTIMA!."
305 PRINT"▮OTRO"
310 PRINT"▮JUEGO?S/N"
315 GETA$:IFA$="S"THENPOKE54276,0:POKE54276,17:GOSUB505:GOTO95
320 IFA$<>"N"THENGOTO315
325 GOSUB505:PRINT"SQQQ▮▮ADIOS!QQQQQ"
330 END
335 I=-1:PRINT"SQQ▮ YO ME"
340 PRINT"▮ LLAMO"
345 PRINT"▮ SIMON"
350 GOSUB505
355 PRINT"SQQ▮ QUIERES"
360 PRINT"▮ JUGAR MI"
365 PRINT"▮JUEGO?S/N"
370 GETA$:IFA$<>"N"ANDA$<>"S"THEN370
375 IFA$="N"THEN485
380 IFA$="Y"THENGOSUB505
```

PROGRAMA COMPROBADO commodore Magazine

```
385 PRINT"SQQ■REPITE   "
390 PRINT"■   LA     "
395 PRINT"■SECUENCIA"
400 GOSUB505
405 PRINT"SQQ■DE COLOR"
410 PRINT"■     Y "
415 PRINT"■ SONIDO."
420 GOSUB505
425 PRINT"SQQ■ CADA "
430 PRINT"■ VEZ HAY "
435 PRINT"■  MAS ."
440 GOSUB505
445 PRINT"SQQ■  DEBES "
450 PRINT"■ ESTAR "
455 PRINT"■ ATENTO!"
460 GOSUB505
465 PRINT"■ESPERA"
470 GOSUB505
475 GOSUB505
480 RETURN
485 GOSUB 505
490 PRINT"SQQ■ ADIOS "
495 PRINT"■ENTONCES!QQQQQQ"
500 END
505 IFI=-1THENFORZ=1TO2000:NEXT:POKE54276,0:POKE54276,33:POKE54273,50
510 PRINT"SQQ■         "
515 PRINT"■            "
520 PRINT"■            "
525 IFI=-1THENPOKE54273,0
530 RETURN
535 DATA0,0,0,255,128,1,255,192,3,255,224,2,0,32,2,0,32,2,54,32,2,99,32
540 DATA2,99,32,14,193,184,10,8,40,10,8,40,8,24,8,8,193,136,8,99,8,8,28,8
545 DATA4,0,16,4,0,16,7,128,240,5,255,208,2,231,32,99
550 DATA2,255,128,10,190,160,42,255,168,170,255,170,174,235,186,174,235,186
555 DATA174,235,186,174,235,186,46,170,184,43,170,232,10,235,160,10,235,160
560 DATA10,190,160,10,195,160,3,60,192,3,255,192,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

Cómo diseñar juegos para ordenador (capitulo 5)



En el capítulo anterior vimos cómo se creaba una aventura básica incluyendo los componentes esenciales: diversas habitaciones, diversos objetos y un monstruo que nos atacaba si nos quedábamos en la habitación (para aquellos que no lo hayan descubierto todavía diremos que si dejan la caja de las galletas en la misma habitación que el monstruo, éste no les ataca), pero esta estructura es muy simple y fácil de aprender. Naturalmente se puede complicar añadiendo más habitaciones y más monstruos, pero hay otros métodos relativamente sencillos de mejorar el juego dándole un aspecto nuevo y más sabroso.

CREANDO UN CARACTER

El jugador del ejemplo del capítulo anterior era muy simple, podía estar vivo o muerto sin ninguna transición intermedia y sin ninguna característica real. Una persona real, por otro lado, tiene una gran cantidad de características que influyen en su actuación, puede estar herida o cansada, puede ser muy lista o ser tonta, etc. Todos estos datos hacen que a veces logre hacer una cosa y otras no pueda (excepto en las películas en las que el bueno siempre logra sus objetivos). El personaje que nosotros creemos puede tener parte de estas características. Las más importantes y el modo en que se modifican se incluyen a continuación:

EXPERIENCIA: característica que influye en las peleas y otras acciones de difícil realización. Se incrementa cada vez que tenemos que realizar alguna.

INTELIGENCIA: esta es una cualidad fijada desde el principio del juego y no se modifica posteriormente. Nos conviene tenerla cuando comerciamos o discutimos con alguien, ya que influye en el resultado.

FUERZA: la fuerza sirve para luchar y para desplazarse de un stio a otro y disminuye cada vez que realizamos alguna de estas acciones. Si la fuerza llega a cero, el juego debe terminar (muerte por anemia) y se incrementa con las comidas, bebidas y demás sustancias nutritivas (por ejemplo una porción mágica).

BONDAD: En muchos juegos hay un parámetro denominado bondad o amistad que indica la facilidad que tenemos para hacer amigos y por tanto para que éstos nos ayuden. El modo de incrementar la bondad es hacer amigos y disminuye cada vez que atacamos a alguien.

Todos estos parámetros son variables que pueden contener desde un valor mínimo hasta un máximo. Estos límites están fijados desde un principio (por ejemplo mínimo: 0 y máximo: 10) y el valor en cada momento del juego puede variar entre ambos, pero no excederlos (si cuando tenemos una fuerza de diez tomamos comida, el programa debe obedecer la orden, pero no incrementar la fuerza). Cada vez que participemos en una acción que modifique uno de estos datos deberemos hacerlo en función de la acción, el tomar comida incrementa la fuerza en uno y la bebida en 0,5 (si la comida está envenenada puede bajar la fuerza a dos) y cuando hagamos una acción que dependa de otros datos deberemos utilizar lo que se denomina una función de peso. Esta función evalúa todos los factores que intervienen dando porcentaje de intervención a cada uno de ellos y hallando la suma total, esta suma se compara con el valor mínimo necesario para lograr nuestro objetivo y el programa obra en consecuencia. En el caso de una lucha podemos suponer que la fuerza tiene un peso de 60; la experiencia, 30, y la inteligencia, diez. La función de peso correspondiente sería la fórmula

PESO = 0.1 * INTELIGENCIA + 0.3 * EXPERIENCIA + 0.6 * FUERZA

Si cada uno de los datos variase entre cero y diez, el resultado variaría también entre estos límites y se tiene que comparar con el que le demos al enemigo. Por ejemplo, los dragones son de ocho, las serpientes de cuatro y otras personas, siete (o una puntuación distinta para cada persona).

LABERINTOS

Los laberintos son una constante en el mundo de los juegos. El famoso comecocos lleva incorporado un pequeño laberinto y la mayoría de los juegos los utilizan de uno u otro modo.

A la hora de diseñar uno para incluirlo en nuestra aventura surgen las mismas dudas que vimos con las habitaciones en el programa del capítulo anterior. ¿Lo diseñamos fijo o que cambie cada vez que se juegue? Si lo diseñamos fijo tenemos la ventaja de que basta hacerlo a mano y luego meterle los datos a la máquina como hicimos en el juego del castillo, la desventaja consiste en que una vez que se ha recorrido una vez ya nos lo conocemos. En cambio si lo diseñamos nuevo cuando empieza un juego tenemos la ventaja de que es distinto y desconocido, pero por contra el proceso de generar un laberinto no es tan fácil como parece, hay que tener en cuenta diversos factores, el laberinto debe tener entrada y salida y además deben estar conectadas por, al menos, un camino. Por tanto, no podemos generar un laberinto de forma totalmente aleatoria porque se podrían producir situaciones imposibles de resolver (la salida del laberinto rodeada de paredes de modo que no se pudiese llegar a ella).

Otro problema que se debe resolver es el método que se utiliza para almacenar el laberinto en memoria. Este punto es bastante importante, ya que el tamaño del laberinto está limitado por la cantidad de memoria disponible. Un método para resolver este problema consiste en hacer un laberinto de varios pisos, por lo que sólo sería necesario tener en memoria el piso en que nos encontramos y los demás se van generando según cambiamos de piso. El problema en este caso surge cuando volvemos a un piso en el que ya hemos estado. Para que el juego sea real el laberinto debe tener la misma forma durante todo el juego, por lo que tenemos que almacenar todos los niveles por los que hemos pasado, o al menos unos datos que nos sirvan para reconstruirlos. Para esto conviene echar un vistazo a los sistemas de generación de números aleatorios.

NUMEROS DE AZAR EN ORDENADORES

Los ordenadores no pueden crear números aleatorios. El lector que tenga unos conocimientos mínimos de BASIC estará pensando en este momento que la frase anterior está equivocada, ya que el Spectrum incorpora las órdenes RANDOMIZE y RND que están diseñadas para generar este tipo de números. Pero esto es incorrecto, un número aleatorio, o mejor dicho una sucesión aleatoria de números (ya que no tiene sentido hablar de números aleatorios, sino de sucesiones), es aquella que no puede ser generada por ninguna ecuación matemática y todos los ordenadores crean sus números mediante el uso de ecuaciones matemáticas (muy extrañas pero matemáticas) que pueden ser reproducidas. Aunque parezca un in-





conveniente, esto representa una ventaja en muchos casos, en el apartado anterior veíamos la necesidad de generar una serie aleatoria o mejor dicho, seudoaleatoria, que pudiese ser reproducida. La primera idea que surge es utilizar la función RND del ordenador haciendo previamente RANDOMIZE con la semilla guardada, pero esto no vale porque el primer número sería siempre el mismo, pero luego empezaría a generar números distintos cada vez, ya que el procedimiento que usa internamente la máquina se basa en el contador que utiliza para generar la señal de video *(FRAMES)*, este número no es aleatorio pero sí es extraordinariamente difícil de generar por lo que es preferible recurrir a un generador propio

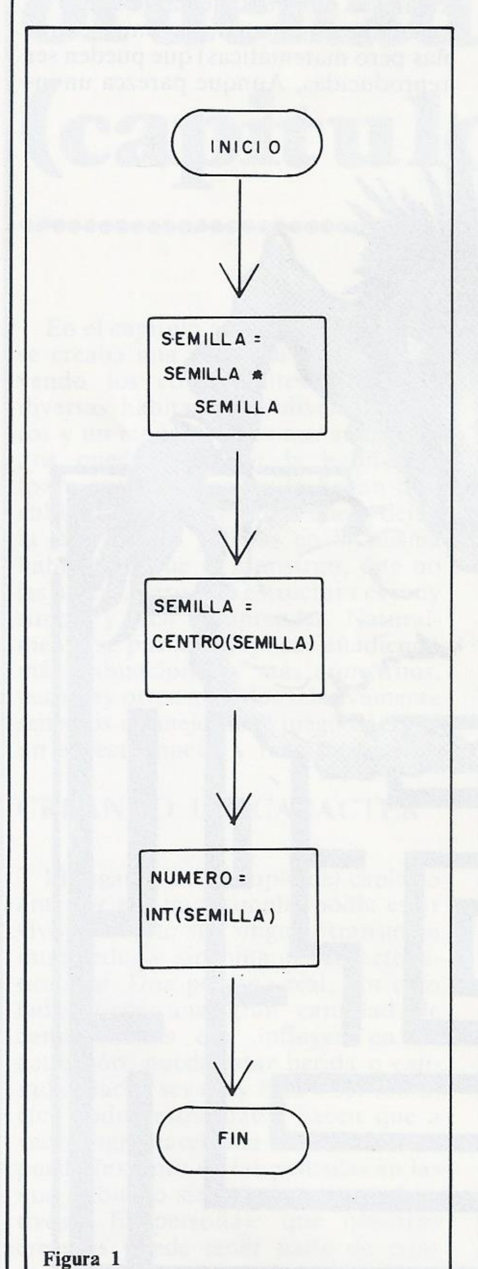


Figura 1

SEMILLA: 0,569982

0,841910688	0,738465148
0,666038826	0,739502621
0,786275343	0,738803844
0,706486250	0,739274584
0,760647571	0,738957504
0,724389733	0,739171100
0,748903972	0,739027222
0,732435524	0,739124141
0,743548017	0,739058856
0,736071531	0,739102833
0,741111773	0,739073210
0,737718448	0,739093165
0,740005059	0,739079723

Figura 2

910	465
038	502
275	803
486	274
647	957
389	171
903	027
435	124
548	058
071	102
111	073
718	093
005	079

Figura 3

sobre el que poseamos más control.

La primera subrutina que creamos se basa en la función coseno, como se ve en el organigrama de la figura 1 se coge la semilla y se calcula su coseno que es el próximo número aleatorio (comprendido entre cero y uno) y, además, es la nueva semilla. Este sistema no es válido porque a los pocos números empieza a generar siempre el mismo como se ve en la tabla 2; este número central es el

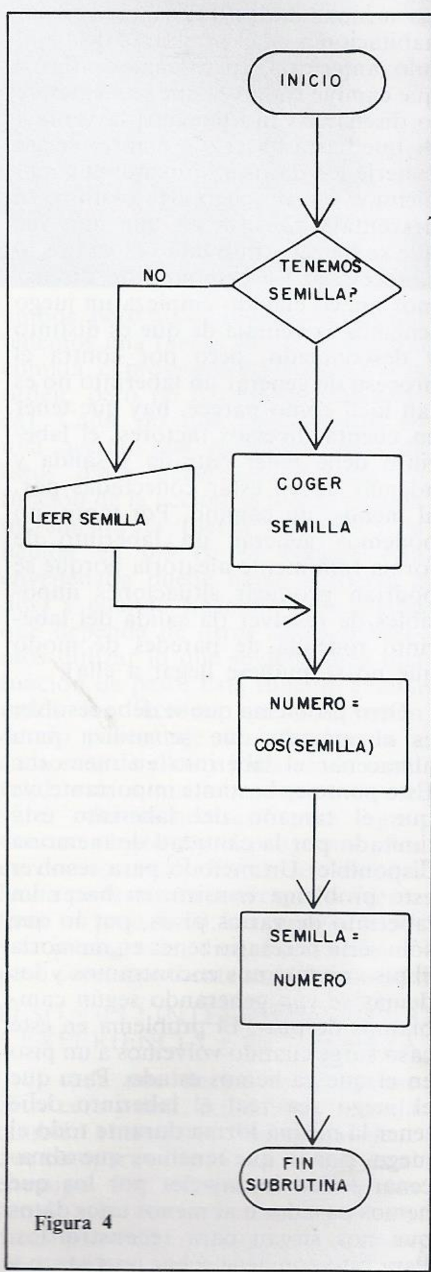


Figura 4

```
10 REM ******************************
20 REM * RUTINA PARA GENERAR NUMEROS *
30 REM *    ALEATORIOS              *
40 REM ******************************
50 REM SE=SEMILLA. LA PRIMERA VEZ QUE
60 REM SE LLAME A ESTA RUTINA SE DEBE
70 REM DAR UN NUMERO ENTRE 0 Y 999 EN
80 REM ESA VARIABLE. EL PROGRAMA LA
90 REM ACTUALIZA CADA VEZ.
100 REM NA= NUMERO ALEATORIO GENERADO
110 REM COMPRENDIDO ENTRE EL 0 Y EL
120 REM 999.
130 SE=(3+SE)↑2
140 SE=SE/1000-INT(SE/1000)
150 SE=SE*1000
160 NA=INT(SE)
170 RETURN
```

432
662
715
399
032
388
174
580
329
623
003
319
963
482
145
301
098
295
673

Figura 6

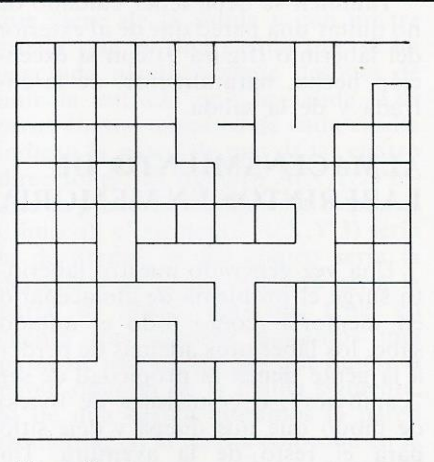


Figura 7

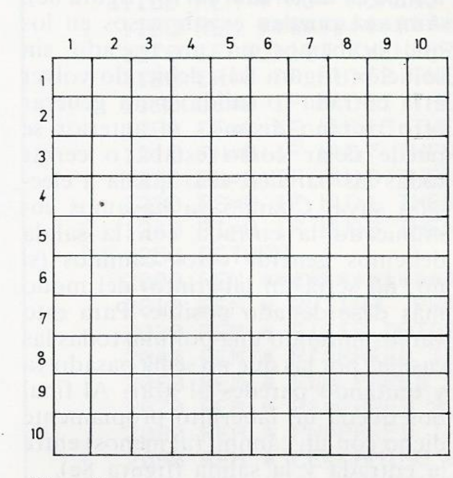


Figura 8-a

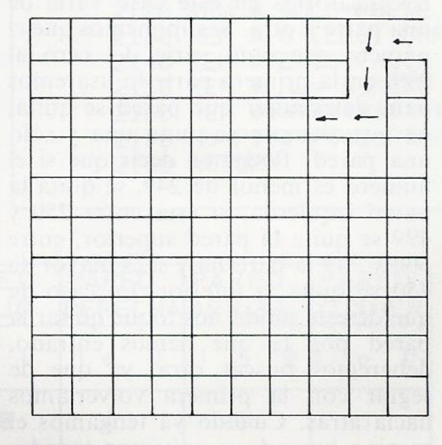


Figura 8-b

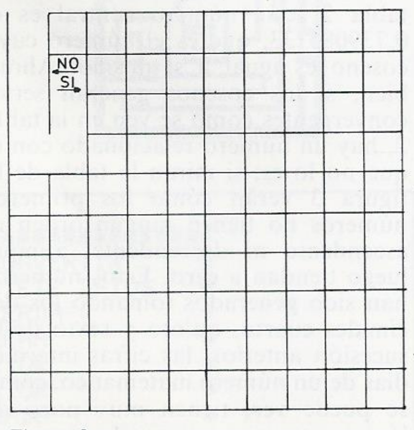


Figura 8-c

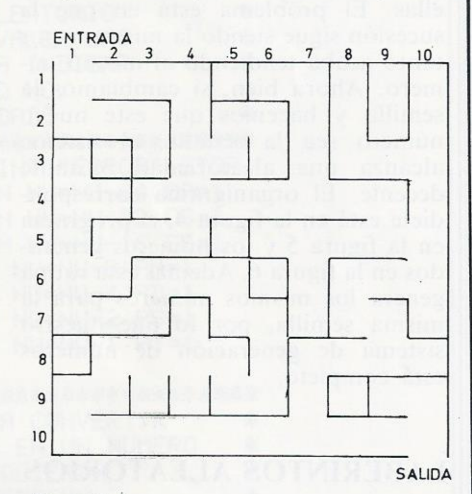


Figura 8-d

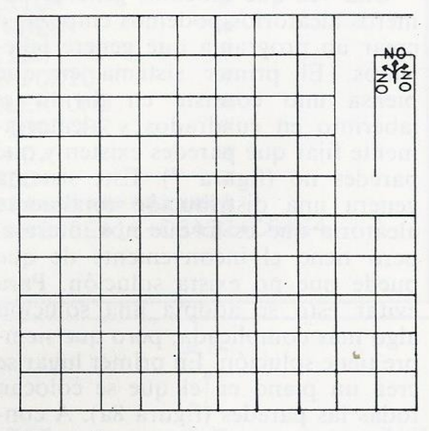


Figura 8-e

0.739085133 que es el número cuyo coseno es igual a sí mismo. Ahora bien, si los cosenos generan series convergentes como se ve en la tabla 2 hay un número relacionado con el que no lo es; si miran la tabla de la figura 3 verán cómo los primeros números no tienen ningún orden ni ascendiente ni descendiente, aunque luego tiendan a cero. Estos números han sido generados tomando los decimales cuarto, quinto y sexto de la

La primera subrutina que creamos se basa en la función coseno, como se ve en el organigrama de la figura 1 se coge la semilla y se calcula su coseno que es el próximo número aleatorio (comprendido entre cero y uno) y, además, es la nueva semilla. Este

sistema no es válido porque a los pocos números empieza a generar siempre el mismo como se ve en la

tabla 2, este número central es el 0.739085133, que es el número cuyo coseno es igual a sí mismo. Ahora bien, si los cosenos generan series convergentes como se ven en la tabla 2, hay un número relacionado con el que no lo es; si miran la tabla de la figura 3 verán cómo los primeros números no tienen ningún orden ni ascendente ni descendente, aunque luego tiendan a cero. Estos números han sido generados tomando los decimales cuarto, quinto y sexto de la sucesión anterior, las cifras intermedias de un número matemático, como se puede ver, tienen muy poco de lógico y no aparentan relación entre ellas. El problema está en que la sucesión sigue siendo la misma y por tanto acaba tendiendo al mismo número. Ahora bien, si cambiamos la semilla y hacemos que este nuevo número sea la semilla, el sistema alcanza una aleatoriedad bastante decente. El organigrama correspondiete está en la figura 4, el programa en la figura 5 y los números generados en la figura 6. Además esta rutina genera los mismos números para la misma semilla, por lo que nuestro sistema de generación de números está completo.

LABERINTOS ALEATORIOS

Una vez que sabemos generar números aleatorios podemos empezar a crear un programa que genere laberintos. El primer sistema en que piensa uno consiste en dividir el laberinto en cuadrados y aleatoriamente fijar qué paredes existen y qué paredes no (figura 7). Este sistema genera una distribución totalmente aleatoria que es lo que nos interesa, pero tiene el inconveniente de que puede que no exista solución. Para evitar esto se adopta una solución algo más complicada, pero que siempre tiene solución. En primer lugar se crea un plano en el que se colocan todas las paredes (figura 8a). A continuación se empiezan a quitar paredes al azar partiendo a la vez desde la puerta de entrada y de salida y quitando paredes de cuadros consecutivos a los que se tenga acceso. Es decir, si en una casilla hemos quitado la pared de la izquierda nos desplazaremos a la casilla de la izquierda y quitaremos otra pared de esa casilla, supongamos que es la pared de arriba (figura 8b), en este caso nos desplazaremos a la casilla superior y quitaremos otra pared. Cada casilla por la que pasemos la marcaremos con un número que será distinto para las que hagamos desde la casilla de entrada y para las de la casilla de salida. Esto tiene dos finalidades, en primer lugar cuando desde una casilla de salida quitemos una pared que dé a una casilla de entrada o viceversa, sabremos que ya hay un camino que une ambas. Por otro lado si desde uno de los dos caminos detectamos un sitio por el que hemos pasado debemos cambiar de dirección para evitar meternos en un círculo sin fin (figura 8c). Aun así pueden existir casos en los que acabemos en un trazado sin solución (figura 8d), debiendo volver a la entrada (o salida) para generar otro camino distinto, el anterior se puede dejar como estaba o cerrar todas las paredes, esto queda a elección suya. Cuando ya hayamos comunicado la entrada con la salida debemos generar otros caminos (si no, no sería un laberinto) del modo más desordenado posible. Para esto vamos mirando una por una todas las casillas por las que no se ha pasado ya y quitando paredes al azar. Al final nos queda un laberinto propiamente dicho con un camino (al menos) entre la entrada y la salida (figura 8e).

El modo en queusamos los números aleatorios en este caso varía de una parte a otra. Si suponemos que el número generado varía del cero al 999, en la primera parte lo usaremos para determinar qué pared se quita, ya que siempre se quita una y sólo una pared. Podemos decir que si el número es menor de 249, se quita la pared izquierda, si está entre 250 y 499 se quita la pared superior, entre 500 y 749 la derecha y si es mayor de 750 se quita la inferior. En caso de que de este modo nos toque quitar la pared por la que hemos entrado, deberemos buscar otra, ya que de seguir con la primera volveríamos hacia atrás. Cuando ya tengamos el camino buscado y quitemos paredes al azar se usa el número aleatorio para ver si quitamos una pared o no, en cada casilla generaremos un número por cada una de las cuatro paredes. Podemos decir que la pared se quita si el número es mayor de 500 y se deja en caso contrario. Si vemos que el programa crea laberintos con demasiados huecos cambiaremos el 500 por un número inferior, si deja demasiadas paredes lo cambiaremos por uno superior.

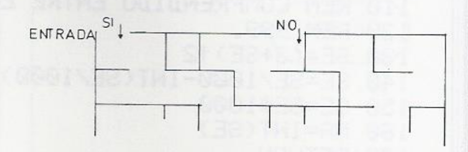


Figura 9

También se debe tener cuidado de no quitar una pared que dé al exterior del laberinto (figura 9) con la excepción hecha, naturalmente, de la entrada y de la salida.

ALMACENAMIENTO DE LABERINTOS EN MEMORIA

Una vez generado nuestro laberinto surge el problema de almacenarlo en memoria (como todo el mundo sabe, los laberintos además de perder a la gente tienen la propiedad de ser "canibytes", o comedores de bytes) de modo que nos quepa y deje sitio para el resto de la aventura. Un sistema básico consiste en una matriz de dos dimensiones, para aquellas personas que no sean expertas en matemáticas se puede hacer una comparación con un tablero de ajedrez o de jugar a los barcos en los que cada casilla se indica por dos números, el primero indica la columna y el segundo la fila, dándonos la situación exacta (figura 10). En este caso podemos definir una matriz con la orden:

```
10 DIM A (10,10,5)
```

En donde: el primer número indica las filas que hay; el segundo, las columnas, y el tercero nos señala cuántos números caben en cada casilla. El primero (a (X,Y.1) siendo X la



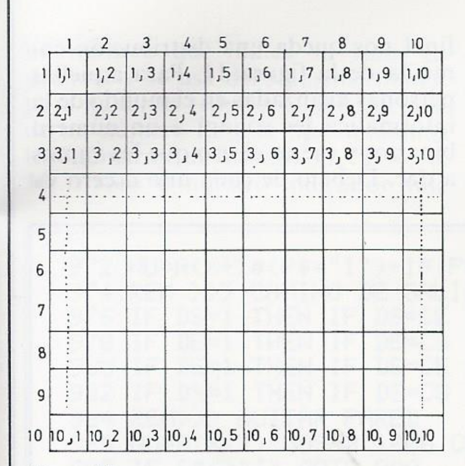


Figura 10

fila e Y la columna) nos dirá si no se ha pasado por esa casilla con un cero, con un uno nos dirá que es una casilla que viene de la entrada, con un dos que viene de la salida y con un tre señala que es una casilla que se ha generado aleatoriamente después de unir la entrada con la salida. Los otros cuatro números de cada casilla indican la pared de una de las cuatro direcciones. El número A(X,Y,2) indica la pared de la izquierda (1 pared. 0 hueco), el elemento A(X,Y,3) sería la de arriba, la del cuatro sería la derecha y el quinto número sería la pared de abajo. Hay que tener en cuenta que cuando quitamos una pared de una casilla hay que quitar la pared opuesta de la casilla siguiente. Si de una casilla quitamos la pared izquierda, en la casilla colocada en la izquierda hay que quitar la pared de la derecha, esto que parece una perogrulla es necesario por el método que se usa para almacenar el laberinto, ya que cada pared está indicada dos veces, una por cada habitación que divide. Si no procedemos así podemos encontrarnos con el curioso caso de que el programa nos deja pasar de una habitación a otra, pero no nos deja volver a la primera (esta opción puede ser interesante en el caso de un laberinto encantado).

Este sistema aunque es sencillo de manejo tiene el problema de que ocupa grandes cantidades de memoria, para intentar reducir este consu-

```
1000 REM ********************************
1010 REM *  RUTINA PARA CONVERTIR       *
1030 REM * UN NUMERO EN SIETE DATOS     *
1040 REM * PI=PARED IZQUIERDA           *
1060 REM * PD= PARED DERECHA            *
1070 REM * PS=PARED SUPERIOR            *
1080 REM * PN=PARED INFERIOR            *
1090 REM * CE=CAMINO ENTRADA            *
1100 REM * CS=CAMINO SALIDA             *
1110 REM * CA=CAMINO ALEATORIO          *
1120 REM * SOLO PUEDE VALER UNO         *
1130 REM * O CERO. PARA INDICAR         *
1140 REM * EXISTENCIA O NO.             *
1150 REM * NU= NUMERO DADO              *
1160 REM ********************************
1170 IF NU>127 THEN PRINT "ERROR":STOP
1180 CA=0:IF NU>63 THEN NU=NU-64:CA=1
1190 CS=0:IF NU>31 THEN NU=NU-32:CS=1
1200 CE=0:IF NU>15 THEN NU=NU-16:CE=1
1210 PN=0:IF NU>7 THEN NU=NU-8:PN=1
1220 PD=0:IF NU>3 THEN NU=NU-4:PD=1
1230 PS=0:IF NU>1 THEN NU=NU-2:PS=1
1240 PI=0:IF NU>0 THEN NU=NU-1:PI=1
1250 RETURN
2000 REM ********************************
2010 REM * RUTINA  PARA CONVERTIR       *
2030 REM * SIETE DATOS EN UN NUMERO     *
2040 REM * PI=PARED IZQUIERDA           *
2060 REM * PD= PARED DERECHA            *
2070 REM * PS=PARED SUPERIOR            *
2080 REM * PN=PARED INFERIOR            *
2090 REM * CE=CAMINO ENTRADA            *
2100 REM * CS=CAMINO SALIDA             *
2110 REM * CA=CAMINO ALEATORIO          *
2120 REM * SOLO PUEDE VALER UNO         *
2130 REM * O CERO. PARA INDICAR         *
2140 REM * EXISTENCIA O NO.             *
2150 REM * NU= NUMERO DADO              *
2160 REM ********************************
2170 NU=PI*1+PS*2+PD*4+PN*8+CE*16+CS*32+CA*64
2180 RETURN
```

Figura 11

CAMINO AZAR	CAMINO SALIDA	CAMINO ENTRADA	PARED ABAJO	PARED DERECHA	PARED ARRIBA	PARED IZQUIER.
1 o 0	1 o 0	1 o 0	1 o 0	1 o 0	1 o 0	1 o 0
64	32	16	8	4	2	1

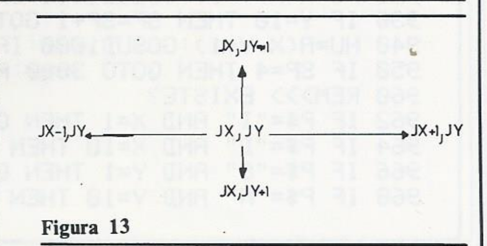


Figura 13

mo vamos a recurrir a unos cuantos "trucos". Si nos fijamos, necesitamos solamente siete indicadores: uno por cada una de las cuatro paredes y uno por cada una de las siguientes condiciones: que venga de la entrada, que venga de la salida o que sea generado al azar. Estos tres últimos son excluyentes (no puede existir una casilla que venga a la vez de la entrada y de la salida) y se pueden representar también con un uno o con un cero. Al final nos queda una distribución como la de la figura 11. Para aquellas personas avanzadas en el mundo de la informática les sonará a un número binario, y ése es el uso que les vamos a dar. Debajo de cada uno o cero va

```
5 GOTO 3000
10 REM ******************************
20 REM * RUTINA PARA GENERAR NUMEROS *
30 REM *     ALEATORIOS             *
40 REM ******************************
50 REM SE=SEMILLA. LA PRIMERA VEZ QUE
60 REM SE LLAME A ESTA RUTINA SE DEBE
70 REM DAR UN NUMERO ENTRE 0 Y 999 EN
80 REM ESA VARIABLE. EL PROGRAMA LA
90 REM ACTUALIZA CADA VEZ.
100 REM NA= NUMERO ALEATORIO GENERADO
110 REM COMPRENDIDO ENTRE EL 0 Y EL
120 REM 999.
130 SE=(3+SE)↑2
140 SE=SE/1000-INT(SE/1000)
150 SE=SE*1000
160 NA=INT(SE)
170 RETURN
500 REM ******************************
510 REM * PONER INDICADORES A CERO   *
520 REM ******************************
530 REM
540 PI=0:PS=0:PD=0:PN=0:CS=0:CE=0:CA=0
550 RETURN
600 REM ******************************
610 REM *    PONER LAS CUATRO        *
620 REM *       PAREDES              *
630 REM ******************************
640 PI=1:PD=1:PS=1:PN=1:RETURN
800 REM >>> COLOCAR CASILLA ENTRADA O SALIDA
810 GOSUB 10
820 IF NA<250 THEN P$="I":GOTO 860
830 IF NA<500 THEN P$="S":GOTO 860
840 IF NA<750 THEN P$="D":GOTO 860
850 P$="N"
860 REM >>>VER CASILLA IZQUIERDA
865 SP=0
870 IF X=1 THEN SP=SP+1:GOTO 890
880 NU=A(X-1,Y):GOSUB1000:IF((CE=1)AND(DE=CE))OR((CS=1)AND(DS=CS))THENSP=SP+1
890 IF X=10 THEN SP=SP+1:GOTO 910
900 NU=A(X+1,Y):GOSUB1000:IF((CE=1)AND(DE=CE))OR((CS=1)AND(DS=CS))THENSP=SP+1
910 IF Y=1 THEN SP=SP+1:GOTO 930
920 NU=A(X,Y-1):GOSUB1000:IF((CE=1)AND(DE=CE))OR((CS=1)AND(DS=CS))THENSP=SP+1
930 IF Y=10 THEN SP=SP+1:GOTO 950
940 NU=A(X,Y+1):GOSUB1000:IF((CE=1)AND(DE=CE))OR((CS=1)AND(DS=CS))THENSP=SP+1
950 IF SP=4 THEN GOTO 3080:REM CAMINO BLOQUEADO. VOLVER A EMPEZAR
960 REM>>> EXISTE?
962 IF P$="I" AND X=1 THEN GOTO 800
964 IF P$="D" AND X=10 THEN GOTO 800
966 IF P$="S" AND Y=1 THEN GOTO 800
968 IF P$="N" AND Y=10 THEN GOTO 800
```

un número que indica su valor en decimal y se usa del siguiente modo: para hallar la representación decimal de un número de este tipo sumamos los números de debajo (1, 2, 4, 8, 16...) sólo si nuestro indicador vale uno, si vale cero no se suma. al final nos queda un solo número que nos indica las siete condiciones que queríamos controlar. Por ejemplo el número siete es la suma 1 + 4, que significa pared izquierda y pared derecha. Si está pensado que se puede representar 1 + 2 + 2 fíjese que cada número sólo puede figurar una vez y en este caso se repite el dos. Si tenemos un número que no figura en la tabla (el 6, por ejemplo) lo debe-

```
972 NU=A(X+1*(P$="I")-1*(P$="D"),Y+1*(P$="S")-1*(P$="N")):GOSUB 1000
974 REM >>> CAMINO DE SALIDA?
976 IF DS=1 THEN IF DS=CE THEN GOTO 4000
978 IF DE=1 THEN IF DE=CS THEN GOTO 4000
980 IF DE=1 THEN IF DE=CE THEN GOTO 800
982 IF DS=1 THEN IF DS=CS THEN GOTO 800
984 REM>>> QUITAR PARED
986 NU=A(X,Y):GOSUB 1000:CE=DE:CS=DS
987 IF P$<>"I" GOTO 989
988 PI=0:GOSUB2000:A(X,Y)=NU:X=X-1:NU=A(X,Y):GOSUB1000:PD=0:GOSUB2000:A(X,Y)=NU
989 IF P$<>"S" GOTO 991
990 PS=0:GOSUB2000:A(X,Y)=NU:Y=Y-1:NU=A(X,Y):GOSUB1000:PN=0:GOSUB2000:A(X,Y)=NU
991 IF P$<>"D" GOTO 993
992 PD=0:GOSUB2000:A(X,Y)=NU:X=X+1:NU=A(X,Y):GOSUB1000:PI=0:GOSUB2000:A(X,Y)=NU
993 IF P$<>"N" GOTO 996
994 PN=0:GOSUB2000:A(X,Y)=NU:Y=Y+1:NU=A(X,Y):GOSUB1000:PS=0:GOSUB2000:A(X,Y)=NU
996 RETURN
1000 REM ********************************
1010 REM *  RUTINA PARA CONVERTIR      *
1030 REM * UN NUMERO EN SIETE DATOS    *
1040 REM * PI=PARED IZQUIERDA          *
1060 REM * PD= PARED DERECHA           *
1070 REM * PS=PARED SUPERIOR           *
1080 REM * PN=PARED INFERIOR           *
1090 REM * CE=CAMINO ENTRADA           *
1100 REM * CS=CAMINO SALIDA            *
1110 REM * CA=CAMINO ALEATORIO         *
1120 REM * SOLO PUEDE VALER UNO        *
1130 REM * O CERO. PARA INDICAR        *
1140 REM * EXISTENCIA O NO.            *
1150 REM * NU= NUMERO DADO             *
1160 REM ********************************
1170 IF NU>127 THEN PRINT "ERROR":STOP
1180 CA=0:IF NU>63 THEN NU=NU-64:CA=1
1190 CS=0:IF NU>31 THEN NU=NU-32:CS=1
1200 CE=0:IF NU>15 THEN NU=NU-16:CE=1
1210 PN=0:IF NU>7 THEN NU=NU-8:PN=1
1220 PD=0:IF NU>3 THEN NU=NU-4:PD=1
1230 PS=0:IF NU>1 THEN NU=NU-2:PS=1
1240 PI=0:IF NU>0 THEN NU=NU-1:PI=1
1250 RETURN
2000 REM ********************************
2010 REM * RUTINA PARA CONVERTIR       *
2030 REM * SIETE DATOS EN UN NUMERO    *
2040 REM * PI=PARED IZQUIERDA          *
2060 REM * PD= PARED DERECHA           *
2070 REM * PS=PARED SUPERIOR           *
2080 REM * PN=PARED INFERIOR           *
2090 REM * CE=CAMINO ENTRADA           *
2100 REM * CS=CAMINO SALIDA            *
2110 REM * CA=CAMINO ALEATORIO         *
2120 REM * SOLO PUEDE VALER UNO        *
```

```
2130 REM * O CERO. PARA INDICAR       *
2140 REM * EXISTENCIA O NO.           *
2150 REM * NU= NUMERO DADO            *
2160 REM ******************************
2170 NU=PI*1+PS*2+PD*4+PN*8+CE*16+CS*32+CA*64
2180 RETURN
3000 REM ******************************
3010 REM *  PROGRAMA PRINCIPAL        *
3020 REM ******************************
3030 REM
3035 INPUT"SEMILLA ";SE
3040 REM ******************************
3050 REM * GENERAR LABERINTO          *
3060 REM ******************************
3070 DIM A(10,10)
3080 REM>>> PONER PAREDES
3090 FOR X=1 TO 10:FOR Y=1 TO 10
3100 GOSUB 500:GOSUB 600:GOSUB 2000:A(X,Y)=NU
3110 NEXT Y:NEXT X
3120 REM >>> COLOCAR ENTRADA
3130 EX=1:EY=1:GOSUB 500:GOSUB 600:PS=0:GOSUB 2000:A(EX,EY)=NU
3135 REM>>> COLOCAR SALIDA
3140 SX=10:SY=10:GOSUB 500:GOSUB 600:PN=0:GOSUB 2000:A(SX,SY)=NU
3150 X=EX:Y=EY:DE=1:DS=0
3160 GOSUB 800
3170 EX=X:EY=Y
3180 X=SX:Y=SY:DE=0:DS=1
3190 GOSUB 800
3200 SX=X:SY=Y
3210 GOTO 3150
4000 REM>>>SE ENCUENTRAN LOS CAMINOS
4005 IF P$<>"I" GOTO 4015
4010 PI=0:GOSUB2000:A(X,Y)=NU:X=X-1:NU=A(X,Y):GOSUB1000:PD=0:GOSUB2000:A(X,Y)=NU
4015 IF P$<>"S" GOTO 4025
4020 PS=0:GOSUB2000:A(X,Y)=NU:Y=Y-1:NU=A(X,Y):GOSUB1000:PN=0:GOSUB2000:A(X,Y)=NU
4025 IF P$<>"D" GOTO 4035
4030 PD=0:GOSUB2000:A(X,Y)=NU:X=X+1:NU=A(X,Y):GOSUB1000:PI=0:GOSUB2000:A(X,Y)=NU
4035 IF P$<>"N" GOTO 4050
4040 PN=0:GOSUB2000:A(X,Y)=NU:Y=Y+1:NU=A(X,Y):GOSUB1000:PS=0:GOSUB2000:A(X,Y)=NU
4050 REM>>> RELLENAR LAS DEMAS CASILLAS
4060 FOR X=1 TO 10:FOR Y=1 TO 10
4070 NU=A(X,Y):GOSUB1000
4075 REM>>> OCUPADA?
4080 IF CE OR CS THEN GOTO 4200
4090 REM>>> QUITAR PAREDES
4098 GOSUB 10:  IF X<2 OR NA>500 GOTO 4108
4100 NU=A(X,Y):GOSUB1000:PI=0:GOSUB 2000:A(X,Y)=NU
4105 NU=A(X-1,Y):GOSUB 1000:PD=0:GOSUB2000:A(X-1,Y)=NU
4108 GOSUB 10:IF Y<2 OR NA>500 GOTO 4118
4110 NU=A(X,Y):GOSUB1000:PS=0:GOSUB 2000:A(X,Y)=NU
4115 NU=A(X,Y-1):GOSUB 1000:PN=0:GOSUB2000:A(X,Y-1)=NU
4118 GOSUB 10:IF X>9 OR NA>500 GOTO 4128
4120 NU=A(X,Y):GOSUB1000:PD=0:GOSUB 2000:A(X,Y)=NU
4125 NU=A(X+1,Y):GOSUB 1000:PI=0:GOSUB2000:A(X+1,Y)=NU
4128 GOSUB 10:IF Y>9 OR NA>500 GOTO 4200
4130 NU=A(X,Y):GOSUB1000:PN=0:GOSUB 2000:A(X,Y)=NU
4135 NU=A(X,Y+1):GOSUB 1000:PS=0:GOSUB2000:A(X,Y+1)=NU
4200 NEXT Y:NEXT X
```

```
4210 REM ******************************
4220 REM * EMPIEZA EL JUEGO           *
4230 REM ******************************
4240 JX=1:JY=1
4250 NU=A(JX,JY):GOSUB 1000
4260 PRINT "HAY SALIDAS HACIA:"
4270 IF PI=0 THEN PRINT "IZQUIERDA"
4280 IF PS=0 THEN PRINT "ATRAS"
4290 IF PD=0 THEN PRINT "DERECHA"
4300 IF PN=0 THEN PRINT "FRENTE"
4310 PRINT "MOVIMIENTO:"
4320 INPUT A$
4330 A$=LEFT$(A$,1)
4340 REM>>> POSIBLE?
4350 IF A$="I" AND PI=0 THEN JX=JX-1:GOTO 4500
4360 IF A$="A" AND PS=0 THEN JY=JY-1:GOTO 4500
4370 IF A$="D" AND PD=0 THEN JX=JX+1:GOTO 4500
4380 IF A$="F" AND PN=0 THEN JY=JY+1:GOTO 4500
4390 PRINT "NO SE PUEDE":GOTO 4260
4500 IF JY=11 THEN PRINT "SALIO BIEN":STOP
4505 IF JY=0  THEN PRINT "SALIO MAL":STOP
4510 GOTO 4250
```

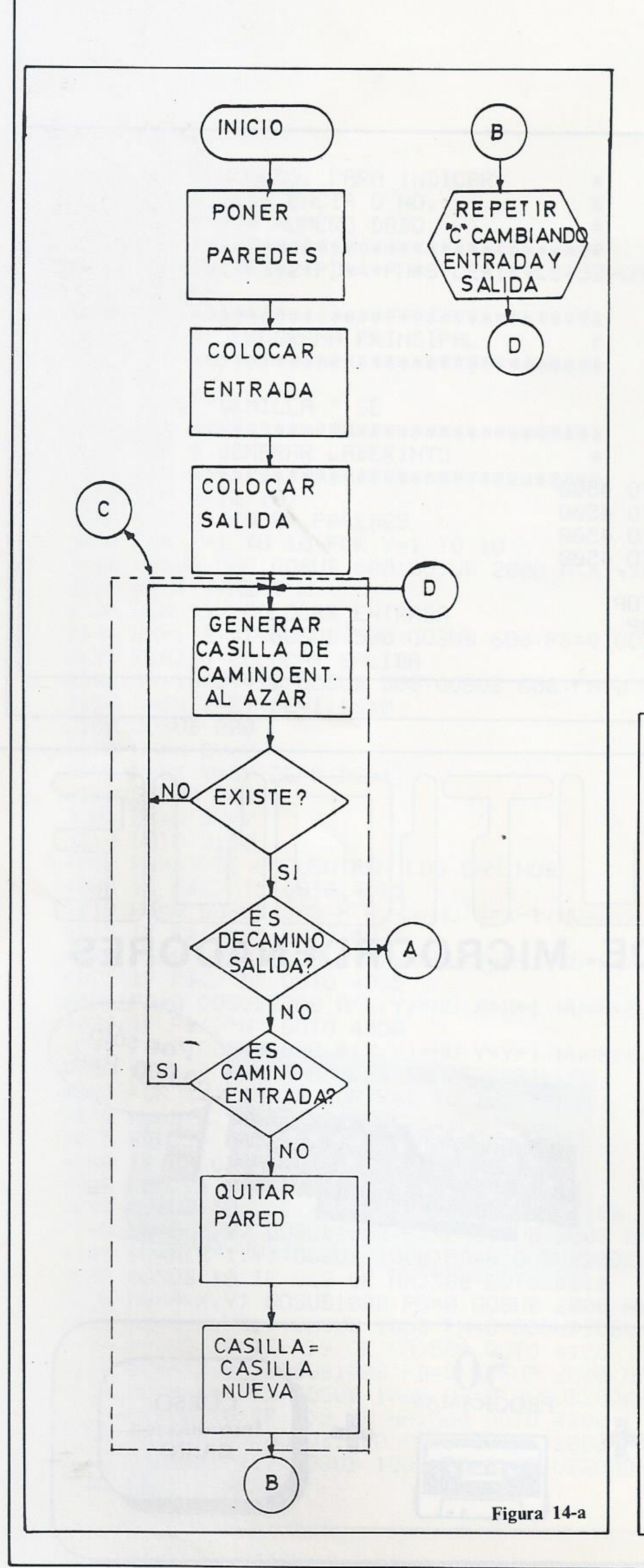


Figura 14-a

TABLA 1

SEMILLA:
0.569982

0.841910688
0.666038826
0.786275343
0.706486250
0.760647571
0.724389733
0.748903972
0.732435524
0.743548017
0.736071531
0.741111773
0.737718448
0.740005059
0.738465148
0.739502621
0.738803844
0.739274584
0.738957504
0.739171100
0.739027222
0.739124141
0.739058856
0.739102833
0.739073210
0.739093165
0.739079723

mos descomponer en suma de éstos (2 + 4).

Si las explicaciones anteriores le han parecido complicadas, no se preocupe y utilice las rutinas de la figura 12. La primera convierte los siete datos en un número para almacenar en la matriz y la segunda coge este número y lo convierte en los datos separados.

De este último modo podemos almacenar los datos de cada casilla en un sólo número y el espacio necesario para almacenar una matriz se reduce a la QUINTA PARTE. En este caso la matriz la definimos:

DIM A(10,10)

Sin poner el tercer número, ya que al ser un uno el ordenador lo toma por defecto.

MOVIENDOSE POR EL LABERINTO

Una vez generado el laberinto el programa debe movernos por él. Este sistema es sencillo. El jugador tendrá dos contadores (a los que llamaremos JX y JY que indicarán respectivamente la fila y la columna en que nos encontramos. Si le mandamos desplazarse a la izquierda bajará el contador JX en una unidad, si es a la derecha deberemos incrementarlo. Si nos vamos para arriba decrementaremos JY en uno y si es para abajo lo incrementaremos (figura 13).

Debemos comprobar que el movimiento es posible (no hay pared por medio) y ver si estamos en la salida con lo que se acabaría el juego.

Como resumen de todo lo dicho se muestra el listado de un programa que genera un laberinto y permite al jugador desplazarse por él para que intente salir (¡ojo!, el programa es como un laberinto de verdad, no muestra el plano y sólo nos dice los caminos posibles que hay desde la habitación en que nos hallamos).

Fernando García Fernández

SU PROGRAMA PARA CUALQUIER SISTEMA COMMODORE PUEDE HACERLE GANAR 5.000 PTAS.

EL PRESENTE CONCURSO ESTA ABIERTO A TODOS NUESTROS LECTORES Y SU PARTICIPACION E INSCRIPCION ES GRATUITA.
LEA LAS BASES DEL CONCURSO

- NO SE ESTABLECEN LIMITACIONES EN CUANTO A EXTENSION, TEMA ELEGIDO O MODELO DE ORDENADOR
- LOS CONCURSANTES DEBERAN ENVIARNOS A LA DIRECCION QUE FIGURA AL PIE, EL CASSETTE O DISKETTE CONTENIENDO EL PROGRAMA, UNA EXPLICACION DEL MISMO Y, AL SER POSIBLE, UN LISTADO EN PAPEL DE IMPRESORA. SE PODRAN ENVIAR TANTOS PROGRAMAS COMO SE DESEE
- LOS PROGRAMAS, PREVIA SELECCION, SERAN PUBLICADOS EN LA REVISTA, OBTENIENDO TODOS ELLOS 5.000 PTAS.
- LA DECISION SOBRE LA PUBLICACION O NO DE UN PROGRAMA CORRESPONDE UNICAMENTE AL JURADO NOMBRADO AL EFECTO POR "COMMODORE MAGAZINE", SIENDO SU FALLO INAPELABLE
- LOS CRITERIOS DE SELECCION SE BASARAN EN LA CREATIVIDAD DEL TEMA ELEGIDO Y LA ORIGINALIDAD Y/O SENCILLEZ EN EL METODO DE PROGRAMACION GLOBAL
- ENVIAR A:
CONCURSO COMMODORE MAGAZINE



commodore Magazine

C/Bravo Murillo, 377. 28002-Madrid

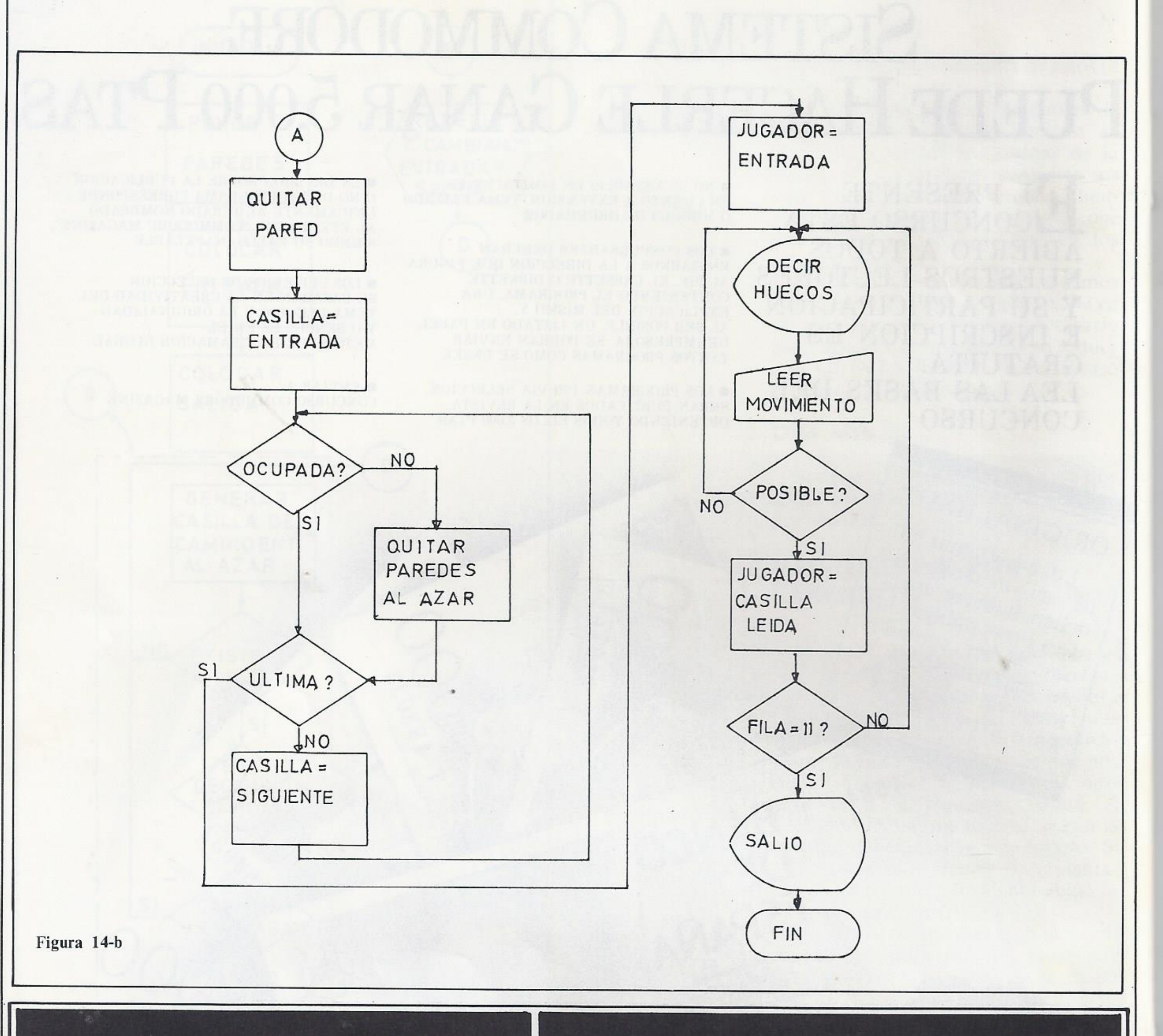

A
QUITAR PARED
CASILLA= ENTRADA
OCUPADA?
NO
SI
QUITAR PAREDES AL AZAR
ULTIMA ?
SI
NO
CASILLA= SIGUIENTE
JUGADOR= ENTRADA
DECIR HUECOS
LEER MOVIMIENTO
POSIBLE?
NO
SI
JUGADOR= CASILLA LEIDA
FILA=11 ?
NO
SI
SALIO
FIN

Figura 14-b



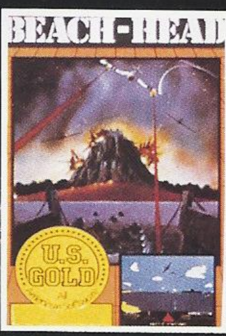
microbyte

PROGRAMAS ORIGINALES DE IMPORTACION PARA

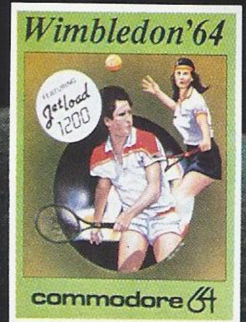
commodore 64



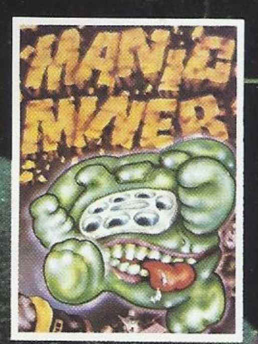
TODOS ESTOS PROGRAMAS HAN ESTADO SITUADOS ENTRE LOS CINCO PRIMEROS PUESTOS DE LAS LISTAS DE SUPERVENTAS BRITANICAS



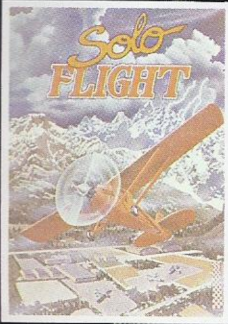
BEACH HEAD. El juego más popular de América con los más sorprendentes efectos de animación (COMP. ANSWER). Los mejores efectos de sonido y gráficos para este computador (U.S. BILLBOARD). Múltiples pantallas, fantásticos efectos tridimensionales en un escenario de acción bélica. Un best-seller (GAMES). 2.300 pts.



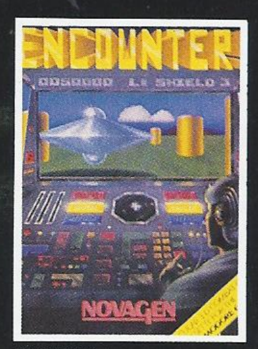
WIMBLEDON 64. Uno de los mejores juegos de simulación de deportes concebido para C-64 (C. & V. GAMES). Horas de entretenimiento para los amantes del tenis y admiradores de McEnroe, Connors y Borg (POP. COMP. WKLY). Los gráficos, la movilidad y el control de las jugadas es fantástico (YOUR 64). 2.300 pts.



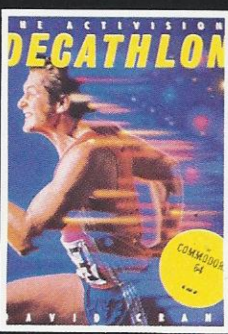
MANIC MINER. Verdaderamente supera a la versión líder de ventas para Spectrum (COM. USGR.). Pantallas y pantallas con todo tipo de gráficos, colores y efectos musicales (HOME. COMP. WKLY). Superior a "Miner 2049". Sin más (CRASH). 1.800 pts.



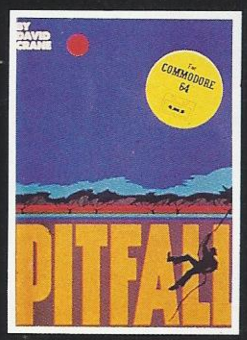
SOLO FLIGHT. Incomparable, sólo necesita pista y alas (COMPUTE). Constantes efectos tridimensionales, perfecto scroll y auténticos efectos sonoros (RUN). 21 aeropuertos diferentes, idénticos instrumentos de bordo, emergencias, V.O.R., una perfecta simulación de vuelo (COM. USER). 3.900 pts



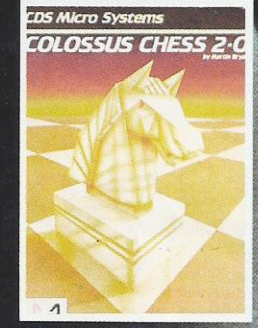
ENCOUNTER. Los gráficos son perfectos, el efecto tridimensional soberbio. Este juego es un "primera clase" (WICH MICRO). ENCOUNTER es el juego de salón más completo escrito para un ordenador personal (GAMES). Nuestra puntuación: un 10 (TPUG). 2.300 pts



DECATHLON. La animación y los gráficos de los atletas es soberbia (P.C. GAMES). Decathlon es una bella simulación de los 10 eventos de la popular prueba olímpica (POPULAR COMPUTING). Uno de los programas más adictivos y espectaculares que he conocido (CRASH). 2.800 pts



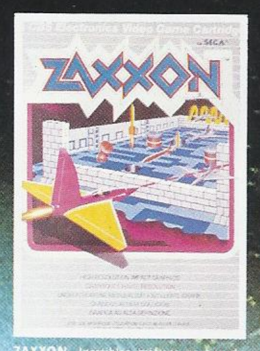
PITFALL. Uno de los juegos para "Atari" más vendidos en U.S.A. ahora en COMMODORE (WHAT MICRO). Caimanes, serpientes, escorpiones, una selva en tu 64 (GAMES COMPUTING). ¿Quién ha sido capaz de encontrar los 32 tesoros escondidos en menos de 20 minutos?... (ELECTRONICS & GAMES). 2.800 pts



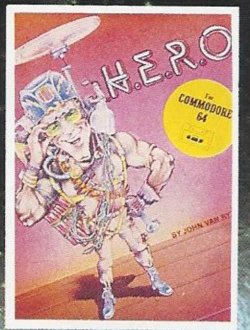
COLOSSUS CHESS 2º. Este juego de inteligencia es indiscutible en vuestra biblioteca (GAMES). El juego mejor logrado en su categoría (POP. COMP. WKLY). Brillante sin duda 4 estrellas (COMP. & VIDEO GAMES). 2.800 pts.



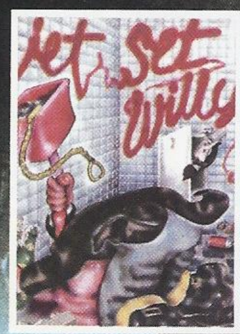
ASTRO CHASE. 34 pantallas con caricaturas animadas en tres dimensiones (GAMES). Juego de fantasía y ciencia ficción del año 1984 (ELECTRONIC GAMES). La mayor innovación en videojuegos (THE WHIZ KID). Este programa batirá todos los records de superventas (RUN). 2.300 pts



ZAXXON. Increíbles gráficos en tres dimensiones con magníficos scrolls diagonales y un supersonido son los ingredientes de este best seller (POP. COMP.). Todas las versiones que han aparecido (Apple, Atari, Coleco, etc.) han sido números uno, sin duda ésta lo será también (U.S. BILLBOARD). 2.300 pts



H.E.R.O. Este programa es el más original de la nueva ola de programas sobre minas y laberintos. El intrépido HERO deberá rescatar a los mineros con su equipo especial de salvamento (ELECTRONIC GAMES). Extraordinarios gráficos y ambientación (HOME COMP. WKLY). 2.800 pts



JET SET WILLY. Excelente, altamente recomendado (GAMES). Color y sonido excelente, gráficos perfectos; nuestra puntuación: un 10 (COMP. & VIDEO GAMES). 1.800 pts

COMMODORE-64

TRON	1.700
FROGGER (Joystick)	1.700
KONG	1.700
SCREEN GRAPHICS	1.800
ENSAMBLADOR	1.900

VIC-20

JOY STICK PAINTER	3, 8, 16K	1.600
PIPER	3, 8, 16K	1.700
TRON	Std	1.600
INVASION	Std	1.600
PHANTON	3, 8, 16K	1.600
PARATROPERS	Std	1.600

Envíenos a **MICROBYTE** San Gerardo, 59 MADRID-35

	Juego	Precio	TOTAL
Nombre			
Apellidos			
Dirección			
Población			
D.P. Teléfono			
Incluyo talón nominativo ☐			
Contra-Reembolso ☐	PRECIO TOTAL PESETAS		
C.M. ENVIOS GRATIS	Pedidos por Teléfono 91-6565002		

CUANDO SE TIENE UN COMMODORE 64 ES MUY DIFICIL SER MODESTO

Cuando se tiene un ordenador personal con 64K de memoria, una magnífica resolución, 16 colores, efectos tridimensionales con "sprites", un sonido equivalente al de un sintetizador, un teclado profesional con 62 caracteres gráficos, toda una amplia gama de periféricos profesionales, la más completa serie de programas educativos, profesionales y de video-juegos...; en resumen, cuando se tiene un ordenador personal como no existe ningún otro en el mercado y el más vendido mundialmente, cuando se tiene el Commodore 64, es muy difícil mostrarlo sin que el orgullo se te note.



EL ORDENADOR PERSONAL DE LA FAMILIA MAS POTENTE

– Sistemas de gestión profesionales series 8000 Y 700. – Ordenador portátil SX 64.
– Ordenador personal COMMODORE 64. – Ordenador familiar VIC 20.



MICROELECTRONICA Y CONTROL, S.A.
c/ Taquígrafo Serra, 7, 5.° BARCELONA-29 c/ Princesa, 47, 3.° G MADRID-8

SLOGAN